DECmate™ II

Word Processing Using Math

digital

AA–N186A–TA

DECmate™ II

Word Processing

Using Math

digital

AA–N186A–TA
First Printing, November 1982

The information in this document is subject to change without notice and should not be construed as a commitment by Digital Equipment Corporation. Digital Equipment Corporation assumes no responsibility for any errors that may appear in this document.

The software described in this document is furnished under a license and may be used or copied only in accordance with the terms of such license.

Digital Equipment Corporation assumes no responsibility for the use or reliability of its software on equipment that is not supplied by DIGITAL.

Copyright © 1982 by Digital Equipment Corporation. All Rights Reserved.

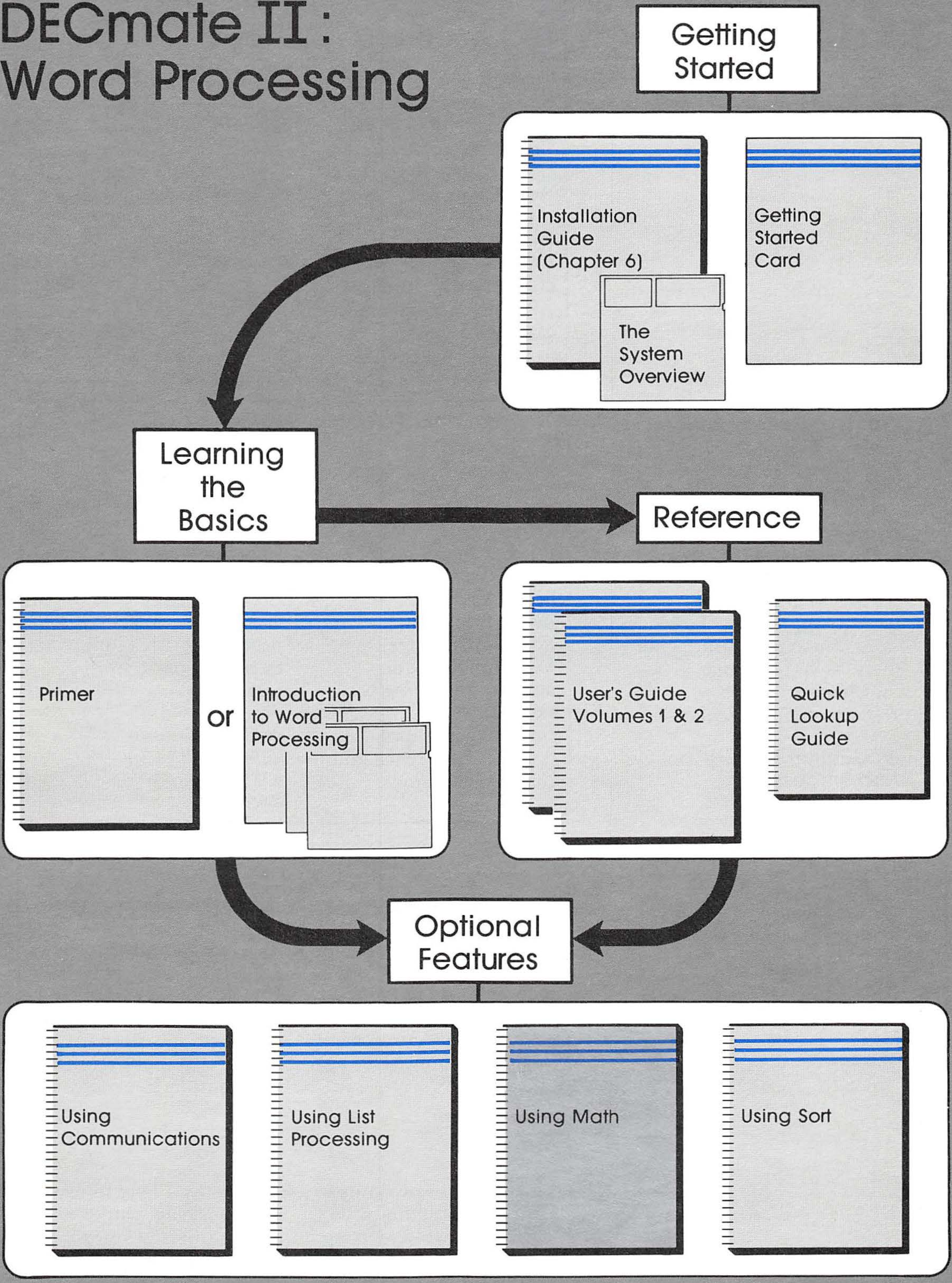
The postage prepaid Reader's Comments Form on the last page of this document requests the user's critical evaluation to assist us in preparing future documentation.

The following are trademarks of Digital Equipment Corporation:

digital™
MASSBUS
PDP
P/OS
Professional
Rainbow
DECwriter
DIBOL
UNIBUS
VAX
VMS
VT
Work Processor
RSTS
RSX
DECmate
DECsystem–10
DECSYSTEM–20
DECUS

Commercial Engineering Publications typeset this manual using DIGITAL's TMS–11 Text Management System.

DECmate II : Word Processing



See the reverse side for a description of each document and diskette.

DECmate II: Word Processing

Learning the Basics

Word Processing Primer

Teaches you word processing basics such as creating, editing, printing and copying documents using a quick, step-by-step format. This manual also tells you where to go next and what to do if you have problems with DECmate.

Introduction to Word Processing

Teaches you basic word processing: how to create, edit and print documents. This self-paced course contains two diskettes (Introduction to WPS–Drive 0 and Introduction to WPS–Drive 1) and a card to get started.

Getting Started

The System Overview

Teaches you how to identify the system parts, and handle diskettes. This self-paced course contains one diskette. Read Chapter 6, Using the System Overview Diskette, in the *DECmate II Installation Guide* (order number EK–DECM2–IN) to begin the course.

Getting Started Card

Teaches you how to turn ON your word processing system, activate features, initialize a document diskette, and back up the system diskette. It also tells you who and when to call for help.

Reference

User's Guide: Volumes 1 & 2

Explains DECmate II word processing procedures. Volume 1 contains basic word processing tasks: how to create, edit and print documents. Volume 2 contains advanced word processing tasks such as pagination, text manipulation, document searches and control commands.

Quick Lookup Guide

Summarizes basic word processing procedures. This guide is organized by tasks, such as creating or printing text. If you are an experienced user, read this guide to review basic word processing.

Note: *You should be familiar with basic word processing procedures before using the optional features.*

Optional Features

Using Communications

Teaches you how to use Communications. Communications lets you send information to and receive information from other computer systems.

Using List Processing

Teaches you how to use List Processing. List Processing lets you print the same document (such as a form letter) many times, varying certain information from copy to copy.

Using Math

Teaches you how to use Math. Math lets you do arithmetic while editing documents or using List Processing.

Using Sort

Teaches you how to use Sort. Sort lets you arrange lists of information in alphabetical or numerical order.

Contents

Preface

Intended Audience vii
How to Use This Manual vii
Conventions viii

1 Overview

What Is Math? 1
Exercises 2
Activating Math 2

2 Activating the Math Feature

Is Math Already Active? 3
To Activate Math 5
If Math Does Not Activate 6

3 Editor Math

Math Areas and Math Control Blocks 9
Interactive Mode 11
Math Formulas 11
Tab Positions and Field Names 13
Allowed Characters 14
Addition and Subtraction 15
  Exercise 1: Addition 15
  Exercise 2: Table with Headings 17
  Exercise 3: Subtraction 19
  Exercise 4: Addition and Subtraction Combined 20
  Exercise 5: Altered Order of Columns 21
  Exercise 6: Negative Numbers 23
  Exercise 7: Constants 24
Multiplication and Division 25
  Exercise 1: Multiplying a Field by a Constant 26
  Exercise 2: Multiplying One Field by Another 27
  Exercise 3: Dividing One Field by Another 28
Totals and Averages 29
  Exercise 1: One-Column Totals 30
  Exercise 2: Multiple Total Columns 32
  Exercise 3: Subtotals 34
  Exercise 4: Averages 38
Parentheses 40
  Exercise 1: Calculation with Parentheses 41
  Exercise 2: The Same Calculation without Parentheses 42
Money and Special Symbols 43
  Standard Format 43
  Format Words 44
  Exercise 1: Money 44
  Exercise 2: Commas 46
  Decimal Fractions 47
  Exercise 3: Asterisks 47
  Negative Numbers 48
  Exercise 4: Trailing Minus Sign 49
  Exercise 5: Parentheses 50
  Exercise 6: Truncation and Rounding Off 52
Calculations for Specific Lines 54
  Exercise: Alternating Calculations 54
Error Messages 57

Summary of Editor Math 63
  Math Areas and Control Blocks 63
  Interactive Mode 63
  Math Commands 64
  Tables 64
  Tab Positions and Field Names 64
  Totaling Tab Positions and Accumulators 64
  Allowed Characters 65
  Writing Equations 65
  Math's Use of Equations 65
  Totals 65
  SET Commands 66
  Averages 66
  Formats 66
  Truncation and Rounding Off 66

4 List Processing Math

Math Commands in the List Document 68
  Starting and Stopping Math 69
Math Formulas in List Processing 70
  Exercise 1: Making a Trial Run 71
  Exercise 2: More than One Control Block 73
Totals and Averages 75
  Exercise: Totals and Averages in the Trailer 76
Changing the Selection 78
  Exercise: Two Reports from One List 78
Error Messages 80
Summary of List Processing Math 87
  Math Commands 87
  Math and List Processing 87
  Field Names 87
  Math's Use of Equations 87
  Totals 88

A Symbol Table

Glossary 91

Index 99

Figures

1 Diskette Maintenance Menu 4
2 Activate Features Menu 4
3 Activate Features Menu with Math Option Active 6
4 Locations of Math Commands in a List Document 70
5 Error Message Screen 81

Preface

PRODUCT NAME: DECmate II/WPS

VERSION: 1.0

Math is an optional feature that you can use with DECmate II word processing software, Version 1.0 or later. Math lets you use your DECmate unit to do arithmetic. It can aid you in making numerical tables while editing or in doing calculations while list processing.

Intended Audience

This manual is written for the user who is familiar with basic word processing on DECmate II or other DIGITAL word processors, and who needs to produce documents with numerical tables or who needs to put calculated figures into list-processed documents.

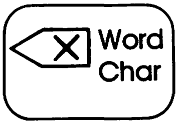
How to Use This Manual

This manual is tutorial in design, so that you can teach yourself to use Math. Chapter 1 gives you an overview of Math. Chapter 2 tells you how to activate the Math feature on your system diskette.

Chapter 3 tells you how to use Math to do arithmetic. It also tells you how to use Math to make numerical tables. The chapter contains exercises in using Math. You should work through these exercises in order before going on to Chapter 4.

Chapter 4 tells you how to combine Math with the List Processing feature. You should be familiar with List Processing before you try to read this chapter. The chapter contains exercises in using Math; work through them in order.

Conventions

`Dot Matrix`	Text printed in dot matrix appears on the screen.
<field name>	Text inside angle brackets is a field name for List Processing, Sort, or Math.
Rubout key	The key marked with a symbol used to rub out functions. 
ALL CAPS	The function or editing key to press. For example: press ADVANCE.
Gold XXX	Press the Gold key and then a function key. For example: Gold MENU.
Type	Enter text into a document from the keyboard.
Press	Enter a command from the keyboard. For example: press DEL WORD.

1

Overview

What Is Math?

Math lets you use your DECmate word processor as a calculator, able to do addition, subtraction, multiplication, and division. You can give the machine an arithmetic problem while you are creating or editing a document. The machine calculates the answer and inserts it in the document at the point you select.

The Math feature includes two kinds of Math — Editor Math and List Processing Math. Editor Math is used to create numerical tables in documents. You supply part of the information on a line of the table, then Math calculates the rest of the information and completes the line for you.

Suppose you are entering a table of test scores. The last entry in each line is the average of all the other entries. With Math, DECmate can calculate these averages and fill in the last entry for you.

Suppose you are entering a table of training expenses. On each line, you enter the number of people taking a given course and the cost per person. Using Math, DECmate can calculate the cost of training all the people taking a given course, then find the total cost of all the training courses.

In addition to doing calculations on separate lines, Editor Math can total up columns, do subtotals, and calculate column averages. Editor Math is discussed in Chapter 3.

List Processing Math is used with the DECmate List Processing feature. It does calculations based on the information in list documents, then inserts the answers in List Processing output documents.

Suppose you use List Processing to produce a price list. Each price has a tax on it. Using Math, you can have DECmate calculate the tax on each price, the total of all the prices both before and after taxes, and the total tax. If you change the list document, DECmate recalculates the taxes and totals automatically.

Suppose you use List Processing to type form letters concerning the interest paid on investments. With Math, DECmate can calculate the interest paid each client as it types the client's form letter.

Like Editor Math, List Processing Math can calculate subtotals and averages in addition to totals. List Processing Math is discussed in Chapter 4.

Exercises

Chapters 3 and 4 contain exercises for you to do as you read through the book. They will help you become familiar with Math. You should do them in order, since the later exercises build on the earlier ones.

Activating Math

To use Math, you must activate it on your system diskette (basic software). If you have just purchased the optional Math feature, go to the next chapter, Activating the Math Feature. You need to activate Math only once on your system diskette. After that, you can use Math whenever you create a document, edit a document, or use List Processing.

2

Activating the Math Feature

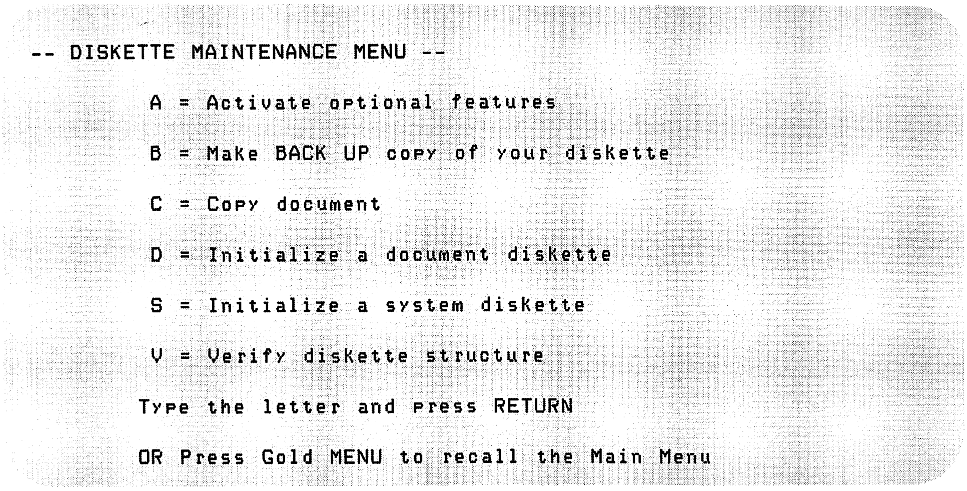
Math is an optional feature that must be activated on your system diskette before it can be used. You need to activate the feature only once on your system diskette.

To activate Math, you need two diskettes: 1) the Math feature diskette (labeled WPS MATH RX50), and 2) a system diskette (labeled WPS THE BASICS RX50).

Is Math Already Active?

Math may already be active on your system diskette. To check whether it is or not, first display the Diskette Maintenance Menu. (See Figure 1.)

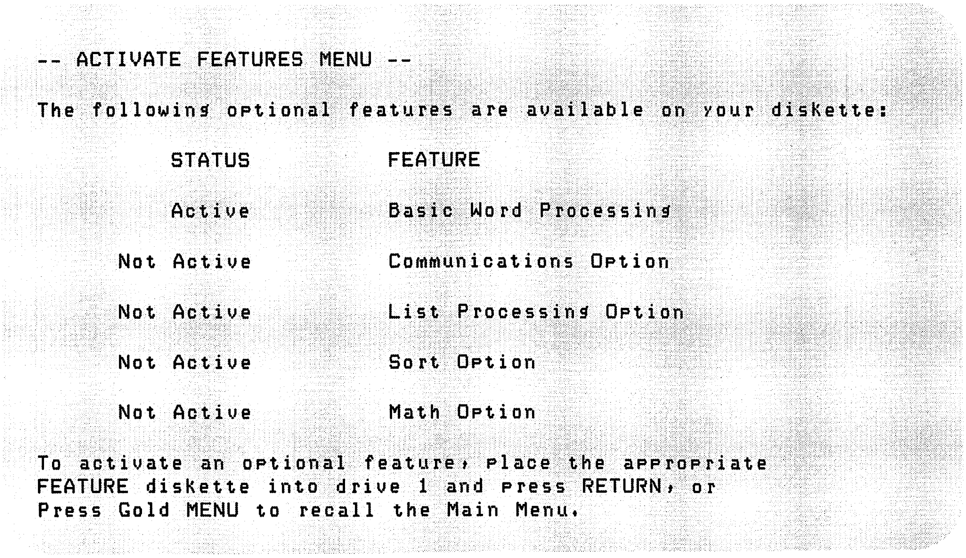
Select MC from the Main Menu.



```
-- DISKETTE MAINTENANCE MENU --

          A = Activate optional features

          B = Make BACK UP copy of your diskette

          C = Copy document

          D = Initialize a document diskette

          S = Initialize a system diskette

          V = Verify diskette structure

         Type the letter and press RETURN

         OR Press Gold MENU to recall the Main Menu
```

Figure 1 Diskette Maintenance Menu

Select the A option. The Activate Features Menu appears. (See Figure 2.)



```
-- ACTIVATE FEATURES MENU --

The following optional features are available on your diskette:

           STATUS              FEATURE

           Active              Basic Word Processing

       Not Active              Communications Option

       Not Active              List Processing Option

       Not Active              Sort Option

       Not Active              Math Option

To activate an optional feature, place the appropriate
FEATURE diskette into drive 1 and press RETURN, or
Press Gold MENU to recall the Main Menu.
```

Figure 2 Activate Features Menu

If *Active* appears opposite *Math Option*, then Math is active. Press Gold Menu to recall the Main Menu. Turn to Chapter 3 for instructions on using Math.

If *Not Active* appears before *Math Option*, you need to activate the feature as explained in the next section.

To Activate Math

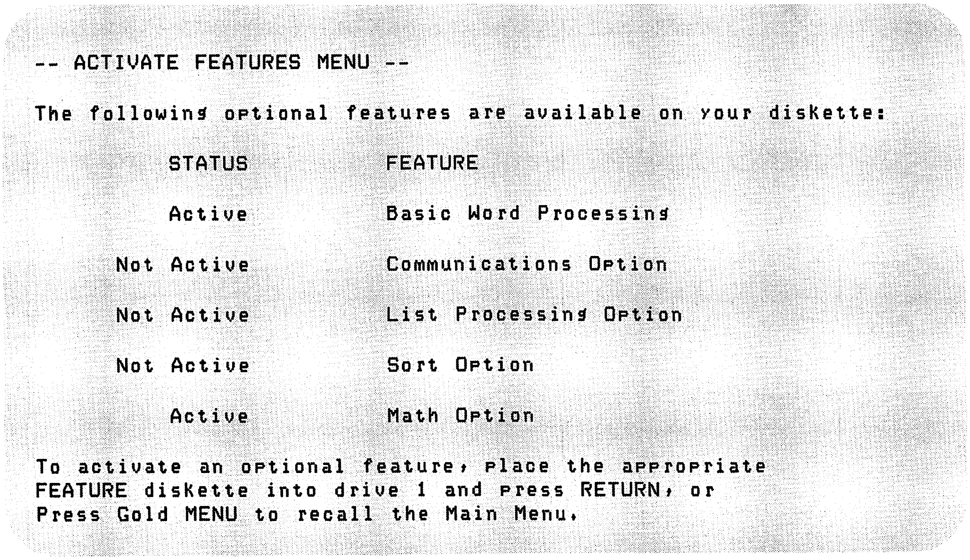
While the Activate Features Menu is displayed:

1. Leave your system diskette in drive 0.
2. Remove any diskette in drive 1.
3. Put the Math feature diskette in drive 1. Make sure the red arrow on the diskette lines up with the red mark on drive 1.
4. Close both drive doors.
5. Press RETURN.

DECmate automatically checks drive 1 to make sure it contains a feature diskette. If it does, DECmate activates the feature. While it is activating the feature, DECmate displays this message:

```
 ---Activating Special Feature Option---

Activation in progress, please stand by...
```

The automatic check and the activating process take less than a minute. During this time, you will hear noises coming from the terminal. When the activating process is complete, the Activate Features Menu reappears showing the Math feature as *Active*. (See Figure 3.)



```
-- ACTIVATE FEATURES MENU --

The following optional features are available on your diskette:

          STATUS              FEATURE

          Active              Basic Word Processing

      Not Active              Communications Option

      Not Active              List Processing Option

      Not Active              Sort Option

          Active              Math Option

To activate an optional feature, place the appropriate
FEATURE diskette into drive 1 and press RETURN, or
Press Gold MENU to recall the Main Menu.
```

Figure 3 **Activate Features Menu with Math Option Active**

Press Gold MENU to return to the Main Menu. Remember to remove the feature diskette from drive 1 before doing word processing.

NOTE: *When you are finished activating your optional features, make a copy of your system diskette.*

If Math Does Not Activate

Check to see which feature diskette you are using. You may have activated another feature.

If Math does not activate, you may see one of these system messages during the activation process.

```
Drive 1 does not contain a valid FEATURE diskette.
```

This means there is an *invalid* feature diskette, such as a damaged feature diskette or an ordinary document diskette, in drive 1.

Press RETURN to recall the Activate Features Menu. Put the correct feature diskette in drive 1 and press RETURN.

```
Error on drive 1 while trying to read block (number).
```

This means that an error developed while the activating process was taking place. The drive door might not be securely closed, or the drive cannot read the feature diskette. Check the drive doors to make sure they are securely closed.

If you still cannot activate the feature, you may have a problem with your terminal. See *DECmate II Owner's Manual* (EK–DECM2–OM).

NOTE: If you use Math with List Processing, then the List Processing feature must also be active on your system diskette.

3

Editor Math

Editor Math helps you make and maintain tables of numbers. When you enter part of the information for one line of the table, Math uses this information to calculate other entries for that line of the table. Math can also do column totals, subtotals, and averages. If you change the table later, Math can repeat its calculations and change the other entries. This keeps the table accurate.

You can also use Editor Math for isolated calculations in a document. Simply treat these as one-line tables.

Math Areas and Math Control Blocks

The DECmate Math feature operates only on numbers it finds inside *Math areas*. A Math area starts with a BEGIN command and finishes with an END command.

These commands, like all other Math commands, must be in *Math control blocks*. To start a Math control block, press Gold CMND, type WPSMATH, and press RETURN. To end the control block, press Gold CMND again. Never put blank lines in a Math control block. Like other control blocks, Math control blocks appear on your screen but are not printed.

To write a BEGIN command, press Gold CMND and type WPSMATH. Press RETURN and type the word BEGIN, press RETURN, then press Gold CMND again. This is a BEGIN command:

```
---------------------- START CONTROL ----------------------
WPSMATH
BEGIN
----------------------- END CONTROL -----------------------
```

The BEGIN command is the signal for DECmate to turn on the Math feature. DECmate turns the feature on when it scrolls past a BEGIN command, or when you enter a BEGIN command.

Math ignores the difference between uppercase and lowercase letters; it makes no difference whether you write *BEGIN* or *begin* or *Begin*. The same is true of all other Math commands.

To end a Math area, use an END command. Press Gold CMND, type WPSMATH, press RETURN, type the word END, press RETURN, then press Gold CMND.

```
---------------------- START CONTROL ----------------------
WPSMATH
END
----------------------- END CONTROL -----------------------
```

An END command must be the only command in its control block. It cannot share the block with any other commands.

The END command is the signal for DECmate to turn Math off. Math also turns off when you use go-to-page, press Gold TOP DOCMT, or press Gold BOT DOCMT.

Moving the cursor back a line turns Math off, if the cursor is inside a Math control block, or if it enters or leaves a Math control block while backing up. To turn Math on again, move forward into a control block containing a BEGIN command.

NOTE: *When you edit a Math control block, back up to a place before the beginning of the Math area and scroll through the area. This makes Math rework the tables using the new commands.*

Interactive Mode

You can prevent Math from turning on by changing the *Math interactive mode* setting on the Editor Menu.

To turn off the interactive mode, display the Editor Menu and type MA NO. Editor Math does not do any calculations. The interactive mode stays off until you turn it back on, or until you file the document.

To turn on the interactive mode, display the Editor Menu and type MA YES. Editor Math turns on when you move the cursor forward into a control block containing a BEGIN command. Math then calculates answers and puts them in the document as soon as you move to the next line. The interactive mode stays on until you turn it off.

Whenever you select a document for editing or create a new document, the interactive mode comes on automatically. Leave the interactive mode on while you do the exercises.

Math Formulas

Once you have turned Math on, you need to tell it what calculations to perform. You do this by writing the calculation out as an equation in a FORMULA command.

To write a FORMULA command, press Gold CMND, type the word WPSMATH, and press RETURN. Type the word FORMULA, followed by at least one space. Then enter the equation you wish the Math feature to use. The next sections discuss the rules for writing these equations. When you have finished the equation, press RETURN to end the FORMULA command.

Do *not* enter a hard return unless you want to begin a new Math command. When Math encounters a hard return, it expects to find a new command or the end of the control block. If it finds anything else, it will give you this error message:

```
ERROR -- Formula in error at this position (unexpected return)
```

Press RETURN and edit the command.

Here is a sample control block of FORMULA commands:

```
----------------------- START CONTROL -----------------------
WPSMATH
FORMULA   <:D1> + <:D2> + <:D3> = <:D4>
FORMULA   <:D1> - <:D3> = <:D5>
FORMULA   <:D3> / <:D4> = <:D6>
------------------------ END CONTROL ------------------------
```

This control block contains three FORMULA commands and their equations. As you see, several different FORMULA commands can share the same Math control block.

The first FORMULA command tells Math to add together the numbers in the first three columns of the table, then write the sum in the fourth column.

Next, the second FORMULA command tells Math to subtract the number in the third column from the number in the first and write the difference in the fifth column.

Finally, the third FORMULA command tells Math to divide the number in the third column by the number in the fourth and write the quotient in the sixth column.

Math always does its calculations in the order in which they appear in the control block.

Math performs these three calculations, in order, on every line of the table, one line at a time. *Math applies every equation in the control block to every line of the table.* If there is enough information, Math calculates the answer and writes it at the position you select.

Math goes on using those equations until it comes to another Math control block or the end of the document. As soon as Math comes to a new Math control block, it stops using the equations of the previous control block. It only uses equations from the new block.

If you like, you can combine the first FORMULA commands in the Math area with the BEGIN command, as in this example:

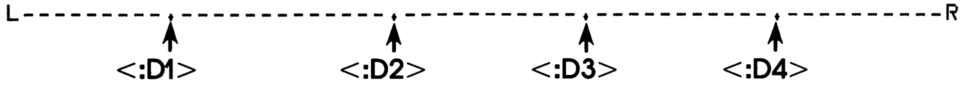
```
----------------------- START CONTROL -----------------------
WPSMATH
BEGIN
FORMULA   <:D1> + <:D2> + <:D3> = <:D4>
FORMULA   <:D1> - <:D3> = <:D5>
FORMULA   <:D3> * <:D4> = <:D6>
------------------------ END CONTROL ------------------------
```

Tab Positions and Field Names

Math looks for numbers at *decimal-aligned tab positions*. It also writes the answers at decimal-aligned tab positions. You must use a ruler with decimal-aligned tabs on any text where you wish to use Math. These tabs are indicated on your ruler by a decimal point.

Math stores the numbers at these tab positions in *fields*. A field is a small section of DECmate's memory. Math uses a field to hold a number.

The field names for these fields always start with an angle bracket, a colon, and capital D,<:**D**. Next comes a number, followed by an angle bracket,>. <:D1>, <:D2>, and <:D3> are typical field names. The number in the field name matches the number of the decimal-aligned tab position, counting from left to right.


L---------.--------------.------------.------------.----------R
<:D1>
<:D2>
<:D3>
<:D4>

You use these field names in the FORMULA commands to tell Math where to find the numbers and where to put the answers.

The maximum number of tab positions depends on the size of your ruler. If it is 80 columns wide or smaller, you can have as many as 18 tabs. If it is wider than 80 columns, you can have as many as 38 tabs.

Allowed Characters

Besides field names, the only characters allowed in Math equations are:

0, 1, 2, 3, 4, 5, 6, 7, 8, 9	Numerals, used to write *constants.* Constants are numbers you write into the equation. They do not change from one line of the table to the next. They are discussed in Exercise 7.
. ,	Decimal point and comma, which can be used to write constants.
$	A dollar sign, indicating money.
+	A plus sign, indicating addition or positive numbers.
–	A minus sign, indicating subtraction or negative numbers.
*	An asterisk, indicating multiplication.
/	A slash, indicating division.
( )	Parentheses, used to group operations together. The parentheses must always match; for every) there must be exactly one (.
=	An equals sign, indicating the end of the calculation. Every equation must have exactly one = sign in it. You are not allowed to put +, –, *, /, (, or) on the right side of an = sign. Instead, you must follow the = sign with the name of the place you want the answer entered.
	A space. You can put spaces into the FORMULA command wherever you please. Math ignores them, but they can make the equation easier for you to read. Field names are the only exception; do not put extra spaces in a field name, or Math will not recognize it.

You may make your equation as long as 200 characters. If you go over this limit, Math gives you this error message:

```
ERROR -- Line has exceeded 200 characters.
```

Press RETURN and edit the equation.

If your equation is long, it may not fit on a single line. This is no problem, since the command word wraps to the next line, as would any other text. Math reads both lines as a single command. Do *not* put in any returns until you have finished entering the command.

The following sections contain exercises based on making tables of inventories and sales. They will help you become familiar with Math. **Do the exercises in order** and keep the tables you create; you will use them again in later exercises.

Addition and Subtraction

Exercise 1: Addition.

- Create a document to keep your exercises in.
- Create the following control block:

```
------------------------ START CONTROL -------------------------
WPSMATH
BEGIN
FORMULA   <:D1> + <:D2> = <:D3>
-------------------------- END CONTROL -------------------------
```

- Beneath the control block, create and embed a ruler with three decimal tabs:

```
L--------------.----------------.--------------.---------------R
```

- Press TAB, type 2, press TAB again, type 3, then press RETURN.

Math will put 5 in the third column. Your screen should look like this:

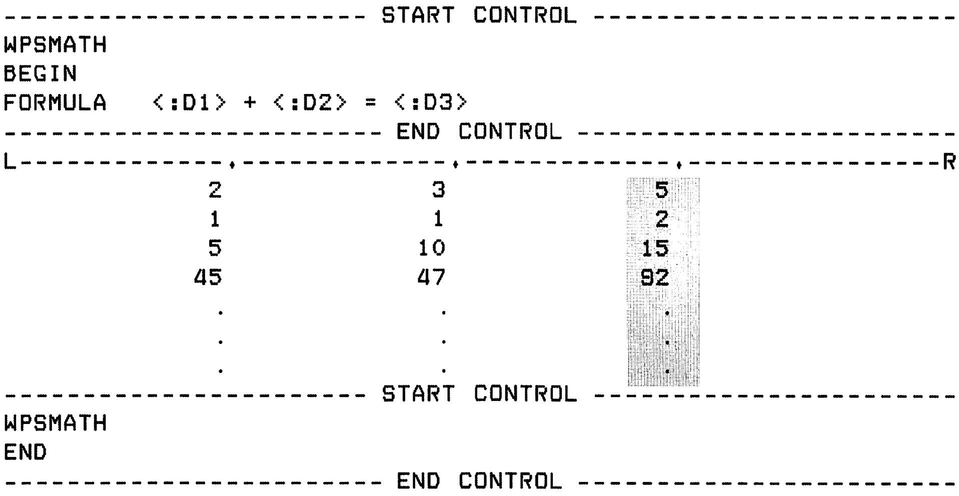
```
------------------------ START CONTROL -------------------------
WPSMATH
BEGIN
FORMULA   <:D1> + <:D2> = <:D3>
-------------------------- END CONTROL -------------------------
L--------------.----------------.--------------.---------------R
              2                3              5
```

- Continue the pattern. Press TAB, type a number, press TAB again, type a second number, then press RETURN. When you press RETURN, the sum of the first two numbers appears in the third column.

NOTE: Math only performs calculations on lines ending with a hard return. If you do not press RETURN at the end of a line, Math will not make any calculations for that line.

- After you have entered several lines, create another control block and enter an END command.

Your screen should look like this:



```
---------------------- START CONTROL ----------------------
WPSMATH
BEGIN
FORMULA   <:D1> + <:D2> = <:D3>
----------------------- END CONTROL -----------------------
L------------.-------------.-------------.---------------R
             2             3             5
             1             1             2
             5            10            15
            45            47            92
             .             .             .
             .             .             .
             .             .             .
---------------------- START CONTROL ----------------------
WPSMATH
END
----------------------- END CONTROL -----------------------
```

Math puts in the numbers in the shaded column.

As you see, you have made a table, with Math entering the third column for you. The FORMULA command in this control block tells Math to add up the numbers at the first two tab positions and write the sum at the third tab position.

If you print out the document, you will not see the control blocks. Only the numbers will appear. The printed output will look like this:

```
 2            3           5
 1            1           2
 5           10          15
45           47          92
 .            .           .
 .            .           .
 .            .           .
```

The table only needs headings to be complete. You can enter the headings as normal text and position them anywhere you like. But remember these two points:

1. Do not use the decimal tabs to position your headings. Math will try to use the headings as numbers, then stop and give you an error message.
2. If text, or numbers in other columns, or the margins, get in the way of the number Math tries to write out, it will stop and write this message at the bottom of the screen:

```
ERROR -- No room for this result: #########
```

The characters ######### represent the number Math tried to write out. Press RETURN, then move the text, or change the ruler, to give Math room to write out the answer.

Exercise 2: Table with Headings. Suppose you know the number of items in stock from last month and the number received. You need to find out the number in stock at the end of this month.

- Create the Math control block shown below. Press Gold CMND, enter the Math commands shown, then press Gold CMND again:

```
----------------------- START CONTROL -----------------------
WPSMATH
BEGIN
FORMULA   <:D1> + <:D2> = <:D3>
------------------------ END CONTROL ------------------------
```

- Make headings for your table, then put decimal-aligned tabs in the ruler:

```
L--------------T--------------T--------------T------------------R
Item           In stock       Received       In stock
               last month     this month     this month

L---------------- .------------ .-------------- .--------------R
```

Notice there are two rulers — one with ordinary tabs for the table headings and one with decimal tabs for the numbers. The headings must have their own set of tabs. If they were aligned on the decimal tabs, they would be read as invalid numbers.

- Enter the first item name (Red), press TAB, enter the number in stock last month (10), press TAB, enter the number received this month (4), and press RETURN.

Math computes the number in stock this month (14) and enters it in the third column. The cursor moves to the next line, ready for you to enter the next item name.

- Follow the example table below, then add your own entries.

```
----------------------- START CONTROL -----------------------
WPSMATH
BEGIN
FORMULA   <:D1> + <:D2> = <:D3>
------------------------ END CONTROL ------------------------
L--------------T--------------T--------------T------------------R
Item           In stock       Received       In stock
               last month     this month     this month

L---------------- .------------ .-------------- .--------------R
Red             10              4              14
Orange           8              3              11
Yellow          45             14              59
                  .              .               .
                  .              .               .
                  .              .               .
----------------------- START CONTROL -----------------------
WPSMATH
END
------------------------ END CONTROL ------------------------
```

If you print out this table, you will see the headings as well as the numbers. It will look like this:

```
Item          In stock          Received        In stock
              last month        this month      this month

Red            10                 4                14
Orange          8                 3                11
Yellow         45                14                59
  .             .                 .                 .
  .             .                 .                 .
  .             .                 .                 .
```

Exercise 3: Subtraction. Suppose items were sold this month, but not received. Now you want to subtract the number sold from the initial number to find how many remain.

- Set up the table as shown below.
- Enter the first two columns of numbers. Math calculates the third column.
- Add your own entries after the line for yellow items.

```
---------------------- START CONTROL ----------------------
WPSMATH
BEGIN
FORMULA   <:D1> - <:D2> = <:D3>
----------------------- END CONTROL -----------------------

L-------------T-------------T-------------T---------------R
Item           In stock      Sold          In stock
               last month    this month  ` this month

L-------------- .------------ .------------ .-----------R
Red            10             2             8
Orange          8             5             3
Yellow         45            15            30
                .             .             .
                .             .             .
                .             .             .
---------------------- START CONTROL ----------------------
WPSMATH
END
----------------------- END CONTROL -----------------------
```

When printed, the table will look like this:

```
Item          In stock       Sold          In stock
              last month     this month    this month

Red            10              2             8
Orange          8              5             3
Yellow         45             15            30
  .             .              .             .
  .             .              .             .
  .             .              .             .
```

Exercise 4: Addition and Subtraction Combined. In this exercise, items were both sold and received in the course of the month. You want Math to add in the number received and subtract the number sold.

- Set up the table as shown below.
- Enter the first three columns of numbers. Math calculates the fourth column.
- Add your own entries after the line for yellow items.

```
----------------------- START CONTROL ----------------------
WPSMATH
BEGIN
FORMULA   <:D1> + <:D2> - <:D3> = <:D4>
------------------------ END CONTROL -----------------------
L---------T--------------T------------T------------T----------R
Item      In stock       Received     Sold         In stock
          last month     this month   this month   this month

L----------- ,----------- ,--------- ,---------- ,--------R
Red        10             4            2            12
Orange      8             3            5             6
Yellow     45            14           15            44
            .             .            .             .
            .             .            .             .
            .             .            .             .
----------------------- START CONTROL ----------------------
WPSMATH
END
------------------------ END CONTROL -----------------------
```

When printed, the table will look like this:

```
Item      In stock      Received      Sold          In stock
          last month    this month    this month    this month

Red        10              4            2           12
Orange      8              3            5            6
Yellow     45             14           15           44
  .         .              .            .            .
  .         .              .            .            .
  .         .              .            .            .
```

Exercise 5: Altered Order of Columns. So far, Math has calculated the values only for the last column of the table. However, Math can put the values in any column, as long as it has all the necessary numbers when the cursor reaches the end of the line.

- Set up the table as shown below. It is like the one in Exercise 4, but its columns are in a different order.

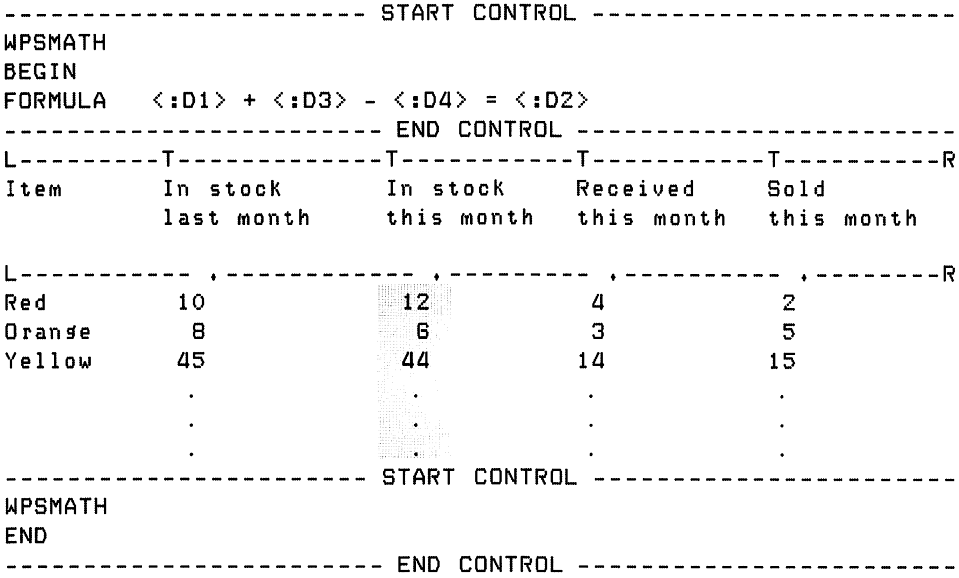
```
----------------------- START CONTROL ----------------------
WPSMATH
BEGIN
FORMULA   <:D1> + <:D3> - <:D4> = <:D2>
------------------------ END CONTROL -----------------------
L---------T-------------T-----------T-----------T---------R
Item      In stock      In stock    Received    Sold
          last month    this month  this month  this month

L----------- ♦------------ ♦--------- ♦---------- ♦--------R
```

Notice that the order of field names in the FORMULA command is different. The new order of field names matches the new order of the headings.

- Enter the first, third, and fourth columns of numbers. Math calculates the second column.
- Start entering data in the table. Enter the item name (Red), press TAB, enter the number in stock last month (10), and press TAB *twice*.

- Enter the number received this month (4), press TAB, enter the number sold this month (2), and press RETURN. The number in stock this month (12) appears in the second column as soon as you go on to the next line.
- Add your own entries after the line for yellow items.

Your table should look like this:



```
----------------------- START CONTROL ----------------------
WPSMATH
BEGIN
FORMULA   <:D1> + <:D3> - <:D4> = <:D2>
------------------------ END CONTROL -----------------------
L---------T-------------T-----------T-----------T----------R
Item      In stock      In stock    Received    Sold
          last month    this month  this month  this month

L---------- ,---------- ,--------- ,---------- ,--------R
Red          10          12          4           2
Orange        8           6          3           5
Yellow       45          44         14          15
              .           .          .           .
              .           .          .           .
              .           .          .           .
----------------------- START CONTROL ----------------------
WPSMATH
END
------------------------ END CONTROL -----------------------
```

When printed, the table will look like this:

```
Item       In stock      In stock      Received      Sold
           last month    this month    this month    this month

Red         10            12             4             2
Orange       8             6             3             5
Yellow      45            44            14            15
  .          .             .             .             .
  .          .             .             .             .
  .          .             .             .             .
```

Exercise 6: Negative Numbers. If necessary, Math gives answers which are *negative numbers*, numbers lower than zero.

In this exercise, Math calculates the change in inventory. Normal, positive numbers mean an increase in inventory; the negative numbers mean a decrease.

- Set up the table as shown below.
- Enter the first and second columns of numbers. Math calculates the third column.
- Add your own entries after the line for yellow items.

```
------------------------ START CONTROL ------------------------
WPSMATH
BEGIN
FORMULA   <:D2> - <:D1> = <:D3>
------------------------- END CONTROL -------------------------
L--------------T--------------T--------------T-----------------R
Item           In stock       In stock       Change in
               last month     this month     inventory

L-----------------.-------------.-------------.----------------R
Red                10             12              2
Orange              8              6             -2
Yellow             45             44             -1
                    .              .              .
                    .              .              .
                    .              .              .
------------------------ START CONTROL ------------------------
WPSMATH
END
------------------------- END CONTROL -------------------------
```

Notice the –2 and the –1 on the second and third lines. This indicates a *decrease* in the number of orange and yellow items.

When printed, the table will look like this:

```
Item          In stock          In stock          Change in
              last month        this month        inventory

Red            10                12                  2
Orange          8                 6                 -2
Yellow         45                44                 -1
  .             .                 .                  .
  .             .                 .                  .
  .             .                 .                  .
```

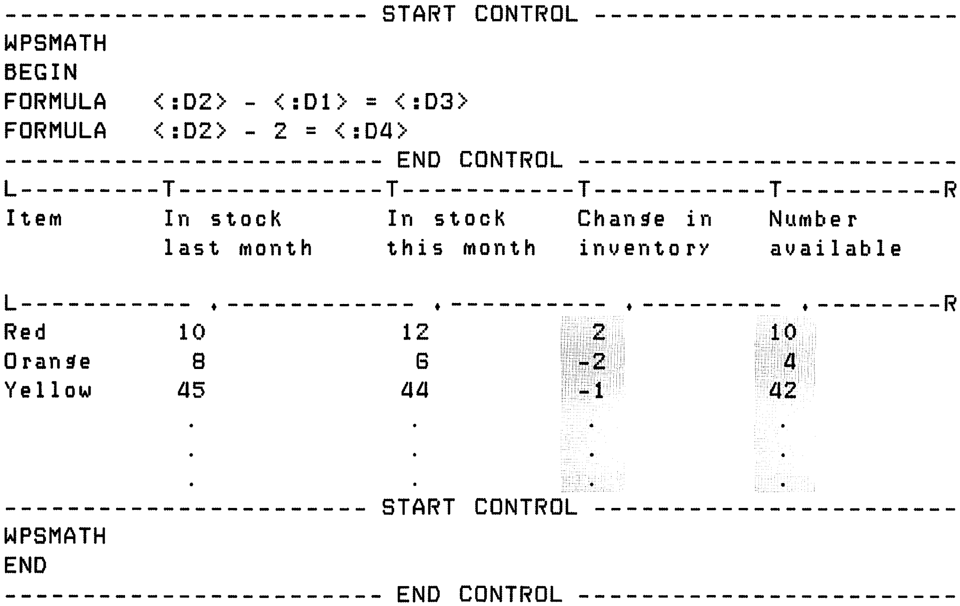
Exercise 7: Constants. A *constant* is any number written in the control block. Since it is part of the control block, its value does not change from one line to the next. You can add, subtract, multiply, or divide by the constant. They can be negative or positive, whole numbers or fractions.

Suppose you must always keep two of each item for display. Those items are not available for sale, even though they are in stock. If you wish to print the number of available items, you can have Math put it in for you. Math takes the number in stock and subtracts two on every line, using this FORMULA command:

```
FORMULA   <:D2> - 2 = <:D4>
```

The constant in this FORMULA command is 2.

- Set up the table as shown below.
- Enter the first two columns of numbers. Math calculates the third and fourth columns.
- Add your own entries after the line for yellow items.



```
------------------------ START CONTROL -----------------------
WPSMATH
BEGIN
FORMULA   <:D2> - <:D1> = <:D3>
FORMULA   <:D2> - 2 = <:D4>
------------------------- END CONTROL ------------------------
L---------T-------------T-----------T-----------T----------R
Item       In stock      In stock    Change in   Number
           last month    this month  inventory   available

L----------- ♦------------ ♦---------- ♦--------- ♦--------R
Red          10            12            2          10
Orange        8             6           -2           4
Yellow       45            44           -1          42
              .             .            .           .
              .             .            .           .
              .             .            .           .
------------------------ START CONTROL -----------------------
WPSMATH
END
------------------------- END CONTROL ------------------------
```

When printed, the table will look like this:

```
Item       In stock     In stock     Change in    Number
           last month   this month   inventory    available

Red          10           12             2           10
Orange        8            6            -2            4
Yellow       45           44            -1           42
  .           .            .             .            .
  .           .            .             .            .
  .           .            .             .            .
```

Multiplication and Division

The exercises in this section are based on grouping the items into sets; items are sold only in sets, not individually.

Exercise 1: Multiplying a Field by a Constant. Suppose you know how many sets were sold and how many items there were in each set. You want Math to calculate how many items were sold. If all the sets have twelve items, then you can have Math multiply the number of sets by a constant (twelve in this case).

- Set up the table as shown below.
- Enter the first column of numbers. Math calculates the second column.
- Add your own entries after the line for yellow items.

```
----------------------- START CONTROL -----------------------
WPSMATH
BEGIN
FORMULA   12 * <:D1> = <:D2>
------------------------ END CONTROL ------------------------
L---------------T---------------T--------------------------R
Item             Number of       Number of
                 sets sold       items sold

L------------------ ,------------- ,---------------------R
Red                   14             168
Orange                10             120
Yellow                12             144
   .                   .              .
   .                   .              .
   .                   .              .
----------------------- START CONTROL -----------------------
WPSMATH
END
------------------------ END CONTROL ------------------------
```

When printed, the table will look like this:

```
Item          Number of         Number of
              sets sold         items sold

Red              14               168
Orange           10               120
Yellow           12               144
  .               .                .
  .               .                .
  .               .                .
```

Exercise 2: Multiplying One Field by Another. In the first exercise, the number of items in each set was always the same. For the second exercise, suppose that the number of items in each set is different, depending on what kind of item it is. (For instance, red items are sold in sets of ten and orange items are sold in sets of twelve.) You cannot use a constant in this case; instead, list the number of items per set under a tab.

- Set up the table as shown below.
- Enter the first two columns of numbers. Math calculates the third column.
- Add your own entries after the line for yellow items.

```
----------------------- START CONTROL ----------------------
WPSMATH
BEGIN
FORMULA   <:D1> * <:D2> = <:D3>
------------------------ END CONTROL -----------------------
L-----------T---------------T-------------T----------------R
Item         Number of        Number of      Number of
             items per set    sets sold      items sold

L--------------- . -------------- . ------------ . ---------R
Red               10               14            140
Orange            12               10            120
Yellow            20                9            180

   .              .                 .              .
   .              .                 .              .
   .              .                 .              .
----------------------- START CONTROL ----------------------
WPSMATH
END
------------------------ END CONTROL -----------------------
```

When printed, the table will look like this:

```
Item          Number of          Number of         Number of
              items per set      sets sold         items sold

Red              10                 14                140
Orange           12                 10                120
Yellow           20                  9                180
  .               .                  .                  .
  .               .                  .                  .
  .               .                  .                  .
```

Exercise 3: Dividing One Field by Another. This exercise is the reverse of the second. Suppose you know how many items of each kind you have in stock, and how many items go into a set of a given kind. You want Math to calculate how many sets you can make from the items in stock. To do this, divide the number of items in stock by the number of items per set.

- Set up the table as shown below.
- Enter the first two columns of numbers. Math calculates the third column.
- Add your own entries after the line for yellow items.

```
----------------------- START CONTROL ----------------------
WPSMATH
BEGIN
FORMULA    <:D1> / <:D2> = <:D3>
------------------------ END CONTROL -----------------------
L---------T-----------------T-----------------T------------R
Item       Number of         Number of         Number of
           items in stock    items per set     sets in stock

L--------------- . ------------- . --------------- . -----R
Red             200                10                20
Orange          144                12                12
Yellow           50                20                 2.5
  .               .                 .                 .
  .               .                 .                 .
  .               .                 .                 .
----------------------- START CONTROL ----------------------
WPSMATH
END
------------------------ END CONTROL -----------------------
```

When printed, the table will look like this:

```
Item          Number of          Number of          Number of
           items in stock     items per set      sets in stock

Red            200                10                 20
Orange         144                12                 12
Yellow          50                20                  2.5
  .              .                 .                  .
  .              .                 .                  .
  .              .                 .                  .
```

Notice the last number in the third line. If the answer to any equation is not a whole number, Math writes out the remainder as a decimal fraction. It carries the number out for as many as six decimal places, if necessary. For example, it writes out .5 for one-half and .333333 for one-third.

In any calculator, division by zero is not allowed. The Math feature is the same. If you should accidentally call for division by zero, Math displays:

```
Divided by zero
```

Edit the control block or the table to remove the division by zero.

Totals and Averages

When you are making up tables of figures, you often want to total up all the numbers in a given column. Math has a special TOTAL command that lets you do this. It is used together with special fields for totals.

A total field, like a decimal tab field, is a small section of memory in which Math stores numbers. Both kinds of fields start with the number zero. At the end of every line in the table, Math sets all the decimal tab fields back to zero, but it leaves the total fields alone.

Total fields have names written <:T1>, <:T2>, <:T3>, and so on. They are just like decimal tab names, except they use a T (for *Total*) instead of a D (for *Decimal*). You must capitalize the letter T.

As with decimal tab names, the number in the name matches the number of the tab position the total will be written in, counting from left to right.

Generally, the starting number in the total field should be zero. If you want the starting number to be any other value, use a SET command. SET commands are discussed in the third exercise of this section.

Every time Math processes a line, you want it to add to the totals that will be written out at the end. You use ordinary FORMULA commands to do this. Here is a sample FORMULA command for calculating a total:

```
FORMULA   <:T1> + <:D1> = <:T1>
```

Notice that <:T1> appears twice, once on each side of the = sign. The <:T1> on the left side of the = sign is the *old* value of the total, before Math adds in <:D1>. The <:T1> on the right side is the *new* value of the total, after <:D1> has been added in. When Math writes out the total, it uses the most recent value of <:T1>.

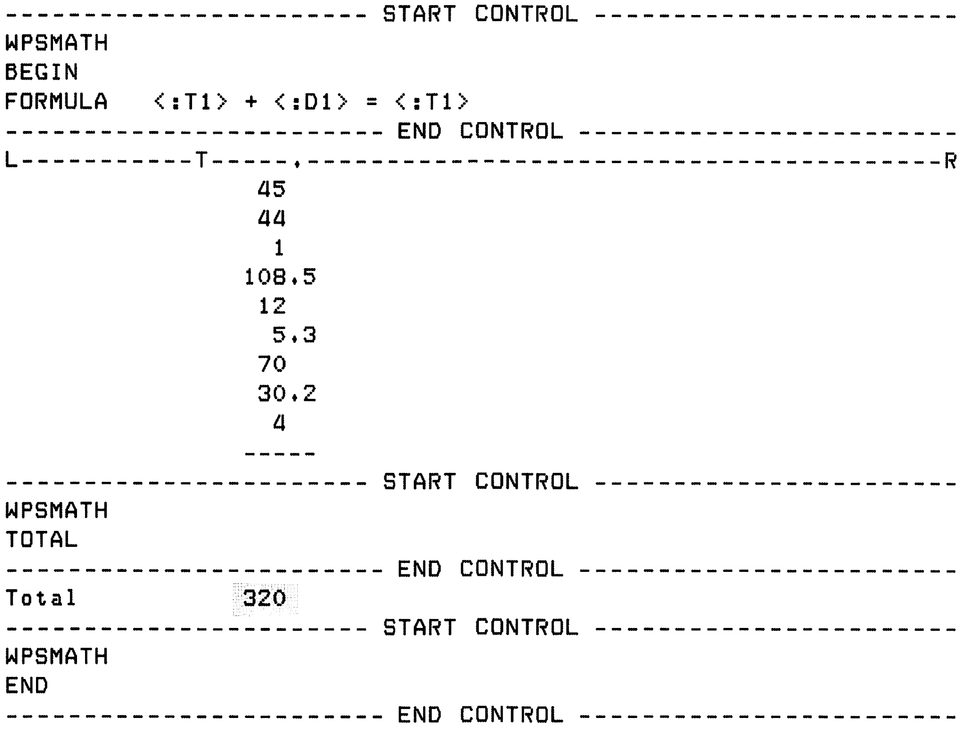
The line on which the totals are written is called the *TOTAL line*. It is the line just below the control block containing the TOTAL command. To write a TOTAL command, just enter the word TOTAL and press RETURN. Here is a sample control block, showing a TOTAL command:

```
---------------------- START CONTROL ----------------------
WPSMATH
TOTAL
----------------------- END CONTROL -----------------------
```

NOTE: *Math adds to the total fields every time it moves forward through a line with numbers on it. If you back up a line while doing totals or averages, Math will add the line a second time when you move forward again. To correct this, back up to a place before the beginning of the Math area and scroll forward. This makes Math rework the totals.*

Exercise 1: One-Column Totals. The simplest total to make is a total of a single column.

- Set up the table as shown below.
- Enter the numbers in the column. Press TAB twice before entering each number. Math writes the sum after the TOTAL command.



```
------------------------ START CONTROL ------------------------
WPSMATH
BEGIN
FORMULA   <:T1> + <:D1> = <:T1>
------------------------- END CONTROL -------------------------
L------------T-----,------------------------------------------R
                  45
                  44
                   1
                 108.5
                  12
                   5.3
                  70
                  30.2
                   4
                 -----
------------------------ START CONTROL ------------------------
WPSMATH
TOTAL
------------------------- END CONTROL -------------------------
Total            320
------------------------ START CONTROL ------------------------
WPSMATH
END
------------------------- END CONTROL -------------------------
```

The total (320, in this case) appears as soon as you move the cursor past the TOTAL line. When you print out the column, it will look like this:

```
                  45
                  44
                   1
                 108.5
                  12
                   5.3
                  70
                  30.2
                   4
                 -----
Total            320
```

Notice the bar at the bottom of the column. It is lined up with a left-justified tab. If it were lined up with the decimal tab, Math would label it as an invalid number. Whenever you put bars under total columns, be sure to position them by using left-justified tabs, right-justified tabs, or spaces. *Do not* use the decimal tabs to position bars.

Be sure to leave space on the TOTAL line, underneath the column. Math always writes the total on the first line following the TOTAL command's control block. If there is text in the way, Math stops and gives you an error message. Arrange space for the answer before you continue.

Exercise 2: Multiple Total Columns. You can create the table in this exercise by editing the table in Exercise 2 of the last section, Multiplying One Field by Another.

- Set up the table as shown below.
- Enter the first two columns of numbers. Make sure each number is lined up under a decimal tab. Math calculates the third column and the column totals.

```
------------------------ START CONTROL ------------------------
WPSMATH
BEGIN
FORMULA   <:D1> * <:D2> = <:D3>
FORMULA   <:D2> + <:T2> = <:T2>
FORMULA   <:D3> + <:T3> = <:T3>
------------------------- END CONTROL -------------------------
L------------T------------------T------------T-----------------R
Item         Number of          Number of    Number of
             items per set      sets sold    items sold

L---------------- .------------- T-- .------- T-- .-----------R
Red             10                 14           140
Orange          12                 10           120
Yellow          20                  9           180
Green           15                  3            45
Blue            10                 11           110
Violet          10                  8            80
------                             --           ---
------------------------ START CONTROL ------------------------
WPSMATH
TOTAL
------------------------- END CONTROL -------------------------
Total                              55           675

------------------------ START CONTROL ------------------------
WPSMATH
END
------------------------- END CONTROL -------------------------
```

Look at the top control block. Notice that there is no <:T1> in the control block. That is because there is no point in making a total of the <:D1> column.

When printed out, the table will look like this:

```
Item        Number of         Number of        Number of
            items per set     sets sold        items sold

Red            10                14               140
Orange         12                10               120
Yellow         20                 9               180
Green          15                 3                45
Blue           10                11               110
Violet         10                 8                80
------                           --               ---
Total                            55               675
```

Exercise 3: Subtotals. Math can calculate subtotals, but you must be careful to keep a record of the grand total at the same time. You can do this by storing it in an *accumulator*. This is a small section of memory in DECmate. You can use it to store numbers that you need to calculate but do not want to write out.

There are only four restrictions on the names you can use for accumulators:

- The name must start with a left angle bracket and colon,<:, and end with a right angle bracket,>.
- The name must not contain more than 30 characters.
- The name must not contain a RETURN.
- The name must not look like a decimal tab field name (such as <:D1>, <:D2>, <:D3>, and so forth).

Except for these restrictions, you can name an accumulator anything you like. <:1>, <:A>, <:total>, and <:sum> are typical accumulator names.

NOTE: *Always enter an accumulator name in exactly the same way, with the same capitalization and spacing. Math treats <:SUM>, <: SUM>, <:Sum>, and <:sum> as completely different accumulators.*

Math will not write out the numbers in an accumulator, unless you are using List Processing. When you are using Editor Math, you must transfer the accumulator's value to a tab field, such as <:D1> or <:T1>. You do this with a *SET command.*

A SET command resembles a FORMULA command. To enter a SET command, type the word SET, enter an equation like the one for a FORMULA command, and press RETURN.

Math performs a SET calculation as soon as you finish the command and press RETURN. Math repeats the SET calculation whenever you move the cursor forward through the SET command.

Compare this with the way Math treats FORMULA commands. Math does not perform a FORMULA calculation until it reaches the table following the control block. It then repeats the FORMULA calculation on every line of the table.

NOTE: Do not scroll through a SET command while Math is working, unless you want Math to perform the SET calculation.

Here are three sample SET commands:

```
SET   <:grand total> + <:T1> = <:grand total>
SET   0 = <:T1>
SET   <:grand total> = <:T1>
```

The following exercise shows three common uses for the SET command: summing subtotals, transferring numbers from one field to another, and resetting total fields.

- Set up the table as shown below.
- Enter the numbers for the colored items. Math calculates the subtotals and the grand total.

```
------------------------ START CONTROL -------------------------
WPSMATH
BEGIN
FORMULA   <:T1> + <:D1> = <:T1>
------------------------- END CONTROL -------------------------
L---T-------------T--,-----------------------------------------R
North
    Red             25
    Green           40
    Blue            10
                   ---
------------------------ START CONTROL -------------------------
WPSMATH
SET   <:grand total> + <:T1> = <:grand total>
TOTAL
------------------------- END CONTROL -------------------------

North total         75
------------------------ START CONTROL -------------------------
WPSMATH
SET   0 = <:T1>
FORMULA   <:T1> + <:D1> = <:T1>
------------------------- END CONTROL -------------------------
South
    Red             15
    Green           35
    Blue            45
------------------------ START CONTROL -------------------------
WPSMATH
SET   <:grand total> + <:T1> = <:grand total>
TOTAL
------------------------- END CONTROL -------------------------
South total         95

------------------------ START CONTROL -------------------------
WPSMATH
SET   <:grand total> = <:T1>
TOTAL
------------------------- END CONTROL -------------------------
L--------------------,-----------------------------------------R
Grand total        170
------------------------ START CONTROL -------------------------
WPSMATH
END
------------------------- END CONTROL -------------------------
```

Notice the SET commands and the accumulator <:grand total>. After Math has calculated each subtotal (North total or South total), a SET command adds the subtotal to <:grand total>. Summing subtotals is one common use of SET.

After the last subtotal is calculated, another SET command transfers the number in <:grand total> to <:T1>. The number is then written out by means of a TOTAL command. This step is necessary because Math cannot write out numbers in accumulators directly. Transferring numbers from one field to another is a second common use of SET.

In the control block before the South list, a SET command resets the value of <:T1> to zero. Setting fields to constants is a third common use of SET.

NOTE: The FORMULA command is repeated before each subtotal. This is because Math cancels all old equations as soon as it comes to a new control block. You must repeat any calculations that you want to continue.

When printed out, the table will look like this:

```
North
   Red            25
   Green          40
   Blue           10
                 ---
North total       75


South
   Red            15
   Green          35
   Blue           45
                 ---
South total       95


Grand total      170
```

Exercise 4: Averages. Go back and look at the example table in Exercise 2, Multiple Column Totals. Suppose you want to find the average number of items sold in a category. There are six in your table, one on each line. You could find the average by dividing <:T3> by six, the number of lines.

When you do not know the number of lines in advance, Math can keep track of this number by using an accumulator. In this exercise, you use the accumulator to store the number of lines, then use this number to calculate the average.

This command adds one to the accumulator every time Math comes to a new line in the table:

```
FORMULA   <:lines> + 1 = <:lines>
```

- Set up the table as shown below.
- You supply the first two columns of numbers. Math calculates the third column, the totals of the second and third columns, and the average of the third column.

```
------------------------ START CONTROL ------------------------
WPSMATH
BEGIN
FORMULA   <:D1> * <:D2> = <:D3>
FORMULA   <:D2> + <:T2> = <:T2>
FORMULA   <:D3> + <:T3> = <:T3>
FORMULA   <:lines> + 1 = <:lines>
------------------------- END CONTROL -------------------------
L---------- T ------------- T ----------- T ------------------R
Item        Number of       Number of     Number of
            items per set   sets sold     items sold

L---------------. -----------T- . -------T-- . --------------R
Red            10             14          140
Orange         12             10          120
Yellow         20              9          180
Green          15              3           45
Blue           10             11          110
Violet         10              8           80
-----                         --          ---
------------------------ START CONTROL ------------------------
WPSMATH
TOTAL
------------------------- END CONTROL -------------------------
Total                         55          675

------------------------ START CONTROL ------------------------
WPSMATH
SET   <:T3> / <:lines> = <:T1>
TOTAL
------------------------- END CONTROL -------------------------
L--------------------------------------------- . -------------R
                                 Average = 112.5
------------------------ START CONTROL ------------------------
WPSMATH
END
------------------------- END CONTROL -------------------------
```

Notice the equation <:T3> / <:lines> = <:T1>, which is the calculation of the average. This equation has to be in the third control block. If it were in the first or second, the <:T3> tab would be used for two things at once — the average and the total. One or the other would have been lost.

When printed out, the table looks like this:

```
Item        Number of         Number of        Number of
            items per set     sets sold        items sold

Red           10                14               140
Orange        12                10               120
Yellow        20                 9               180
Green         15                 3                45
Blue          10                11               110
Violet        10                 8                80
-----                           --               ---
Total                           55               675

                                      Average =  112.5
```

Parentheses

Parentheses are used to group mathematical operations together. Usually, Math performs multiplications and divisions first, then additions and subtractions. But if any of the operations in an equation are inside parentheses, Math always does those before the others. This can make a difference in the answers you get. For example:

<:D1> * (<:D2> + <:D3>) = <:D4>

If <:D1> is 2, <:D2> is 4, and <:D3> is 6, then Math calculates that <:D4> is:

2 * (4 + 6)

2 * 10

20

But if you take away the parentheses, Math does the multiplication first and the addition second. It calculates that <:D4> is:

$$2 * 4 + 6$$

$$8 + 6$$

$$14$$

Exercise 1: Calculation with Parentheses.

- Create the table shown below, listing sets of items sold domestically, sets of items sold abroad, and the total number of items (not sets of items) sold. There are ten items to a set.
- Enter the first two columns. Math calculates the third column.
- Add your own entries after the line for yellow items.

```
------------------------- START CONTROL -------------------------
WPSMATH
BEGIN
FORMULA   (<:D1> + <:D2>) * 10 = <:D3>
-------------------------- END CONTROL --------------------------
L--------------T--------------T--------------T------------------R
Items          Sets sold      Sets sold      Number of
               domestically   abroad         items sold

L-------------------,--------------,--------------,-------------R
Red               80             80          1600
Orange            55            100          1550
Yellow           100             85          1850
  .                .              .              .
  .                .              .              .
  .                .              .              .
------------------------- START CONTROL -------------------------
WPSMATH
END
-------------------------- END CONTROL --------------------------
```

When printed, the table will look like this:

```
Items        Sets sold        Sets sold         Number of
             domestically     abroad            items sold

Red            80               80              1600
Orange         55              100              1550
Yellow        100               85              1850
  .             .                .                .
  .             .                .                .
  .             .                .                .
```

Exercise 2: The Same Calculation Without Parentheses.

- As the second exercise, take the parentheses out of the equation and scroll through the table again.

When printed, the table will look like this:

```
Items        Sets sold        Sets sold         Number of
             domstically      abroad            items sold

Red            80               80               880
Orange         55              100              1055
Yellow        100               85               950
  .             .                .                .
  .             .                .                .
  .             .                .                .
```

Notice that the numbers in the third column are no longer correct.

For more complicated equations, you can put one pair of parentheses inside of another. For example:

100 + ((<:D1> – <:D2>) / (<:D3> – <:D4>)) = <:D5>

In such a case, Math does the innermost operations first. In the example, Math does both subtractions first. Then it divides the first difference by the second. Finally, it adds 100 and enters the answer in <:D5>.

NOTE: *Parentheses always go in pairs. Always put in exactly one) for every (. In every equation, Math checks to make sure that there are as many right parentheses as left ones. If there are not, you get an error message.*

Money and Special Symbols

Standard Format. Unless you tell it otherwise, Math always writes numbers in the same way, called *standard format.* Standard format does not use dollar signs or commas. It does not use decimal points when writing whole numbers. If the number is negative, it puts a minus sign in front of it. If the number is not a whole number, it puts in a decimal point and the digits of the fractional part. It does not put zeros after the last non-zero digit of a decimal fraction. Standard format can handle numbers as long as thirteen digits; as many as six of those digits can come after the decimal point. If the number has more than six places after the decimal point, standard format rounds it off. All the numbers in the exercises so far have been in standard format. Here are some examples of standard format:

Table

Number	Standard Format	
A whole number	53000	
A negative number	–90	
A fraction	0.075	(No trailing zeros)
A nonwhole number	12.5	(No trailing zeros)
The largest possible number	9999999999999	
The smallest possible fraction	0.000001	
The largest possible nonwhole number	999999999999.9	
The largest possible number using all available decimal places	9999999.999999	
0.0000017, rounded off	0.000002	
0.0000012, rounded off	0.000001	

Standard format is useful for many Math applications. But it is not always acceptable. For instance, one-third is written as **0.333333** in standard format. This may be too long to fit into the ruler.

Format Words. If you do not want to use standard format, you can change Math's format with a *format word.* With format words, you can tell Math to write its answers as money, or to limit the number of decimal places in the answers, or to use special symbols.

The format word follows the equation in a SET or FORMULA command. It is put in quotation marks. You can put in spaces for easier reading. Here are some examples:

```
FORMULA   <:D1> / <:D2> = <:D3>      "$,"
FORMULA   <:D1> / <:D2> = <:D3>      "$99,999.99"
FORMULA   <:D1> / <:D2> = <:D3>   "($*, 9999999.99)"
```

Exercise 1: Money. If you want Math to calculate amounts of money, put the format word "$" after an equation. Math will always write a dollar sign, $, in front of the answer to that equation.

If you want Math to round the answer to the nearest penny, you must tell it how many places to expect in the largest answer. For example, if you do not expect any answers as large as one thousand dollars, you would use the format word **"$999.99"**.

Here is a sample control block using money format:

```
---------------------- START CONTROL ----------------------
WPSMATH
FORMULA   <:D1> * <:D2> = <:D3>  "$999.99"
----------------------- END CONTROL -----------------------
```

This format word tells Math to 1) put a dollar sign, $, in front of <:D3> whenever it writes that field; 2) expect no more than three digits in front of the decimal point; and 3) write no more than two digits after the decimal point. If the calculation produces more than two digits, Math will round them off.

The 9s in the format word mark places where Math should expect to write numerals. Pick the highest number of numerals you will want to the right or left of the decimal point, then use that number of **9**s in the format word. Do not use any other numerals in a format word.

- Set up the table as shown below, using money format.
- Enter the first two columns. Math calculates the third column.
- Add your own entries after the line for yellow items.

```
----------------------- START CONTROL -----------------------
WPSMATH
BEGIN
FORMULA   <:D1> * <:D2> = <:D3> "$999.99"
------------------------ END CONTROL ------------------------
L-------------T-------------T-------------T---------------R
Item          Price         Number sold   Sales

L---------------- ,---------------- ,--------- ,-----------R
Red             $1.00               30      $30.00
Orange          $2.25               12      $27.00
Yellow          $0.98              120     $117.60
   .                .                .           .
   .                .                .           .
   .                .                .           .
----------------------- START CONTROL -----------------------
WPSMATH
END
------------------------ END CONTROL ------------------------
```

Notice the numbers in the Price column, which are entered as money. Math accepts the dollar signs in the Price column and treats the money values as ordinary numbers. It writes the answers as money only if there is a dollar sign in the format word.

When printed out, the table will look like this:

```
Item        Price      Number sold      Sales

Red          $1.00         30           $30.00
Orange       $2.25         12           $27.00
Yellow       $0.98        120          $117.60
  .              .          .                .
  .              .          .                .
  .              .          .                .
```

- Now put in a line that will make <:D3> larger than $1000. For example:

```
Brown      $20.00       100
```

Math will write **%2000** in the Sales column. Whenever Math finds that it cannot fit its answer into the format you have given, it writes out a percent sign, %, followed by the answer in standard format.

Exercise 2: Commas. If you deal with very large numbers, commas can make them easier to read. One or more commas in the format word makes Math write a comma every three decimal places to the left of the decimal point. The commas must come after any dollar signs and before the decimal point.

One comma is all you need. These format words all give the same results:

```
"$9,999,999.99"           "$,9999999.99"       "$9999999,,,.99"
```

- Set up the table as shown below. You can do this by editing the table from Exercise 1, so that it looks like the table shown below. Raise the prices and sales, and put commas in the format.

```
----------------------- START CONTROL -----------------------
WPSMATH
BEGIN
FORMULA   <:D1> * <:D2> = <:D3>  "$9,999,999.99"
------------------------ END CONTROL ------------------------
L-------------T--------------T--------------T-----------------R
Item          Price          Number sold    Sales

L-----------------.-----------------.------------.------------R
Red          $2100.00           1500   $3,150,000.00
Orange       $1250.25            512     $640,128.00
Yellow          $8.98         12,000     $107,760.00
  .                .             .           .
  .                .             .           .
  .                .             .           .
----------------------- START CONTROL -----------------------
WPSMATH
END
------------------------ END CONTROL ------------------------
```

Notice the number of yellow items sold. Math accepts numbers with commas in them; it treats them as it would numbers written without commas. But Math puts commas in the answers only if you put a comma in the format word.

When you are not dealing with money, you can use "," alone as the format word. Math will then write the answers in standard format, but with commas put in every three places to the left of the decimal point.

When printed out, the table will look like this:

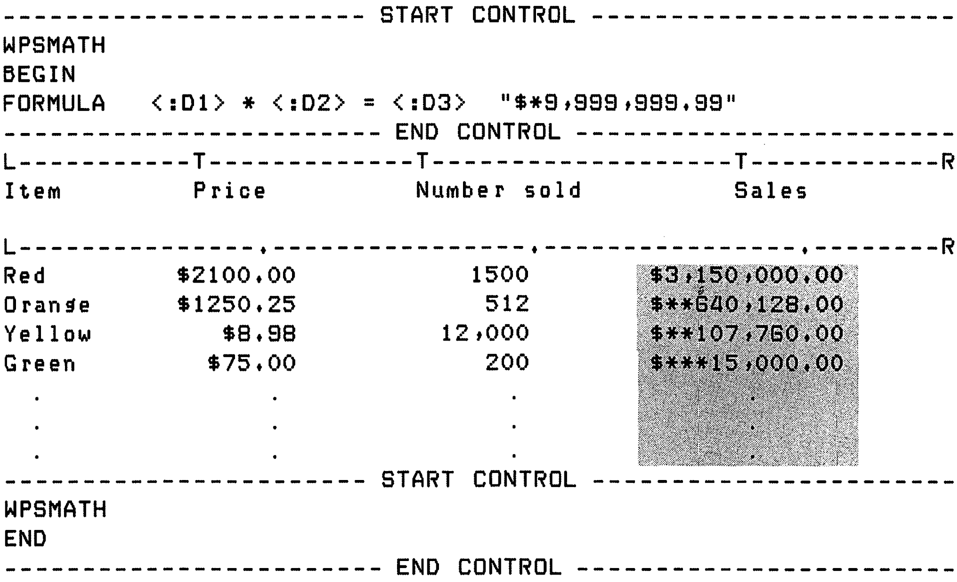
```
Item          Price         Number sold         Sales

Red         $2100.00           1500        $3,150,000.00
Orange      $1250.25            512          $640,128.00
Yellow         $8.98         12,000          $107,760.00
  .              .               .                     .
  .              .               .                     .
  .              .               .                     .
```

Decimal Fractions. In standard format, decimal fractions are written with one zero in front of the decimal point. As many as six decimal places are written out, but trailing zeros are not written. As a result, one-half is written as **0.5**, not **0.500000**.

You can change this format if you wish. If you want to leave off the initial zero, use a format like "**.999999**". However, this format produces *trailing* zeros. With this format, one-half will be written as **.500000**. You must have either trailing zeros or a leading zero.

Exercise 3: Asterisks. Sometimes, you may wish to fill out a column of numbers with asterisks, *, to even out the left edge of the column. To do this, put one asterisk, *, into the format word, after any dollar sign and before any commas or nines. If you use asterisks, you must tell Math how many numerals to expect. You should use an asterisk only in format words with **9**s in them. Put in as many nines as there are numerals in the largest number you expect.

- Set up the table as shown below. You can do this by editing the format word in the table from Exercise 2. Put an asterisk in the format word.
- Add your own entries after the line for green items.



```
------------------------ START CONTROL -----------------------
WPSMATH
BEGIN
FORMULA   <:D1> * <:D2> = <:D3>   "$*9,999,999.99"
------------------------ END CONTROL -------------------------
L-----------T-------------T------------------T------------R
Item         Price         Number sold         Sales

L-------------.-----------------.----------------.--------R
Red          $2100.00           1500      $3,150,000.00
Orange       $1250.25            512      $**640,128.00
Yellow          $8.98         12,000      $**107,760.00
Green          $75.00            200      $***15,000.00
  .               .                .             .
  .               .                .             .
  .               .                .             .
------------------------ START CONTROL -----------------------
WPSMATH
END
------------------------ END CONTROL -------------------------
```

Notice that Math puts asterisks in the place of commas as well as in the place of numerals.

When printed out, the table will look like this:

```
Item        Price        Number sold       Sales

Red         $2100.00       1500        $3,150,000.00
Orange      $1250.25        512        $**640,128.00
Yellow         $8.98     12,000        $**107,760.00
Green         $75.00        200        $***15,000.00
  .              .           .                     .
  .              .           .                     .
  .              .           .                     .
```

Negative Numbers. In standard format, Math indicated negative numbers by a minus sign, –, in front of the number. Two other ways of indicating negative numbers are putting a minus sign *after* the number and putting the number in parentheses, (). You can shift to either of these alternatives by using a format word.

Exercise 4: Trailing Minus Sign. If you want Math to use a trailing minus sign, put one into the format word. It must be the last symbol before the closing quotation mark. With a trailing minus sign, you must tell Math how many numerals to expect in the largest number. That is, you can use trailing minus signs only in format words containing **9**s. Here are some examples of trailing minus format:

```
"$9999.99-"                "999,999-"              "$*99,999.99-"
```

- Set up the table as shown below, with a trailing minus sign in the format word.
- Enter the first two columns. Math calculates the third column.
- Add your own entries after the line for yellow items.

```
---------------------- START CONTROL ----------------------
WPSMATH
BEGIN
FORMULA   <:D2> - <:D1> = <:D3>    "99-"
----------------------- END CONTROL -----------------------
L-------------T-------------T-------------T---------------R
Item           In stock      In stock      Change in
               last month    this month    inventory

L--------------.------------.------------.-------------R
Red            10            12             2
Orange          8             6            2-
Yellow         45            44            1-
  .             .             .              .
  .             .             .              .
  .             .             .              .
---------------------- START CONTROL ----------------------
WPSMATH
END
----------------------- END CONTROL -----------------------
```

When printed, the table will look like this:

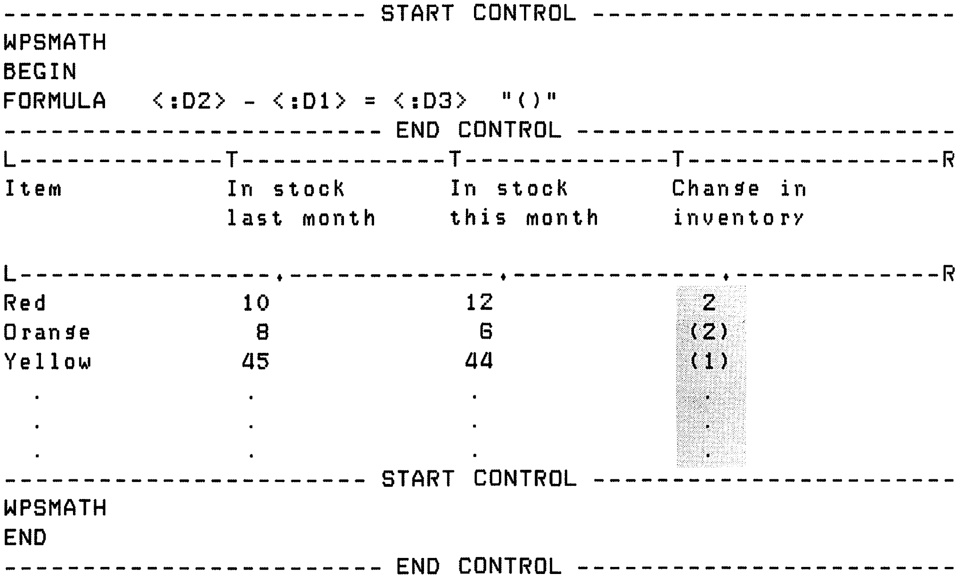
```
Item        In stock      In stock      Change in
            last month    this month    inventory

Red          10            12             2
Orange        8             6            2-
Yellow       45            44            1-
  .           .             .             .
  .           .             .             .
  .           .             .             .
```

Exercise 5: Parentheses. If you want Math to use parentheses, **( )**, put them around the format word. The parentheses should be the first and last symbols in the word, except for the quotation marks. Here are some examples of parenthesis format:

```
"($999.99)"                "()"                "(*,999999)"
```

Notice that you do not have to use 9s for parenthesis format. If your format word is just "**( )**", Math will write its answers in standard format, except that it will use parentheses instead of a leading minus sign to indicate negative numbers.

- Set up the table as shown below. You can do this by editing the format word in the table from Exercise 4.
- Add your own entries after the line for yellow items.



```
------------------------ START CONTROL ------------------------
WPSMATH
BEGIN
FORMULA   <:D2> - <:D1> = <:D3>   "()"
------------------------- END CONTROL -------------------------
L--------------T--------------T--------------T----------------R
Item           In stock       In stock       Change in
               last month     this month     inventory

L----------------,--------------,--------------,--------------R
Red             10             12             2
Orange           8              6            (2)
Yellow          45             44            (1)
  .              .              .             .
  .              .              .             .
  .              .              .             .
------------------------ START CONTROL ------------------------
WPSMATH
END
------------------------- END CONTROL -------------------------
```

When printed, the table will look like this:

```
Item          In stock       In stock       Change in
              last month     this month     inventory

Red            10             12              2
Orange          8              6             (2)
Yellow         45             44             (1)
  .             .              .              .
  .             .              .              .
  .             .              .              .
```

Exercise 6: Truncation and Rounding Off. So far, Math has always dealt with overlong fractions in one way; it has rounded the fraction off before writing it. For instance, if Math were to write 0.796 in money format, with only two decimal places, it would round the number up to 0.80. But sometimes you do not want Math to round off. You simply want it to drop the extra digits. This is called *truncation.* If Math were to truncate 0.796, it would write out 0.79.

You can make Math truncate with a *TRUNCATE command.* You enter this into a control block on a line by itself. This can be any line after the first one, the line holding the word WPSMATH. For example:

```
----------------------- START CONTROL -----------------------
WPSMATH
FORMULA   <:D1> / 10 = <:D3>
TRUNCATE
FORMULA   <:D2> * 10 = <:D4>
------------------------ END CONTROL ------------------------
```

Math truncates all its answers until it comes to a new Math area. In the example above, Math will truncate both <:D3> and <:D4> if it cannot write out their values in full. When Math comes to a new Math area, it will go back to rounding off unless there is a new TRUNCATE command.

If you want to go back to rounding off before the end of the Math area, use a ROUND command. Type ROUND on a line by itself in a control block. All following answers will be rounded off, not truncated.

TRUNCATE and ROUND always have a line to themselves. This can be any line in the control block, after the first. Put in only one of them. If you put in both, Math uses the last one in the block.

- Set up the table as shown below.
- Enter the first and second columns. Math calculates the third and fourth columns.
- Add your own entries after the line for yellow items.

```
----------------------- START CONTROL -----------------------
WPSMATH
BEGIN
FORMULA   <:D1> / 10 = <:D3>  "999"
FORMULA   <:D2> * 10 = <:D4>  "$999.99"
TRUNCATE
------------------------ END CONTROL ------------------------
L-----------T-----------T-----------T-----------T----------R
Item         Number      Price       Number      Price
             of items    of items    of sets     of sets

L--------------.-----------.-----------.-----------.------R
Red            38       $1.08        3          $10.80
Orange         22       $0.75        2           $7.50
Yellow         45       $0.56        4           $5.60
  .             .            .       .               .
  .             .            .       .               .
  .             .            .       .               .
----------------------- START CONTROL -----------------------
WPSMATH
END
------------------------ END CONTROL ------------------------
```

When printed, the table will look like this:

```
Item        Number        Price         Number       Price
            of items      of items      of sets      of sets

Red          38            $1.08         3           $10.80
Orange       22            $0.75         2            $7.50
Yellow       45            $0.56         4            $5.60
  .           .                .         .                .
  .           .                .         .                .
  .           .                .         .                .
```

- Change the TRUNCATE command to a ROUND command.

When printed, the table will look like this:

```
Item       Number      Price       Number        Price
           of items    of items    of sets       of sets

Red         38           $1.08        4           $10.80
Orange      22           $0.75        2            $7.50
Yellow      45           $0.56        5            $5.60
  .          .                .        .                .
  .          .                .        .                .
  .          .                .        .                .
```

Notice that the numbers of sets have gone up by one for Red and Yellow. It looks as if you have more complete sets of those items than you really do.

Calculations for Specific Lines

Sometimes, you may want to perform different calculations on different lines of the same table. But Math always applies *all* its FORMULA commands to every line of the table. However, if you don't want Math to perform a certain calculation on a certain line, you can prevent it by refusing to give Math enough information.

Suppose Math has the equation <:D1> + <:D2> = <:D3> in its control block. If you type in the value for <:D1>, but go to the next line without typing in a value for <:D2>, Math writes in no value for <:D3>. It doesn't have enough information.

Exercise: Alternating Calculations

- Set up the table as shown below. Math uses one equation on the second line of each entry and another equation on the third line.
- On the first line of each entry, type the color and press RETURN. (Type Red for the first entry.)
- On the second line of each entry, type the number made at the first tab position, type the cost per item at the third tab position, then press RETURN. (Type 100 and $21.50 for the first entry.)

Math will calculate the total cost and write it at the fifth tab position.

- On the third line of each entry, type the foreign sales at the second tab position, type the domestic sales at the fourth tab position, then press RETURN. (Type $1570.00 and $1600.00 for the first entry.)

Math will calculate the total sales and write it at the sixth tab position.

- Add your own entries after the lines for yellow items.

```
----------------------- START CONTROL ----------------------
WPSMATH
BEGIN
FORMULA   <:D1> * <:D3> = <:D5> "$99,999.99"
FORMULA   <:D2> + <:D4> = <:D6> "$99,999.99"
------------------------ END CONTROL -----------------------
L---T------------------T--------------------T-----------------R
Item
    Number made        Cost/item           Total cost
      Foreign sales        Domestic sales        Total sales
L--------,--------,---------,--------,---------,------,---R
Red
      100                 $21.50           $2,150.00
             $1570.00             $1600.00         $3,170.00
Orange
      200                 $10.00           $2,000.00
             $1800.00             $1050.00         $2,850.00
Yellow
      240                 $11.50           $2,760.00
             $2055.00             $2070.00         $4,125.00
  .
  .     .                    .                    .
  .     .           .        .         .          .       .
        .           .        .         .          .       .
                    .                  .                  .
----------------------- START CONTROL ----------------------
WPSMATH
END
------------------------ END CONTROL -----------------------
```

Notice that the two FORMULA commands do not share any field names. No field name appears in both equations. This is a necessary part of doing different calculations on different lines.

Never use the same field name in two different equations, if you are doing different calculations on different lines. Otherwise, the results of the first equation will not appear.

When printed out, the table will look like this:

```
Item
   Number made          Cost/item           Total cost
     Foreign sales        Domestic sales      Total sales

Red
      100                 $21.50            $2,150.00
          $1570.00            $1600.00             $3,170.00

Orange
      200                 $10.00            $2,000.00
          $1800.00            $1050.00             $2,850.00

Yellow
      240                 $11.50            $2,760.00
          $2055.00            $2070.00             $4,125.00
  .
  .     .                       .                   .
  .     .         .             .     .             .      .
        .         .             .     .             .      .
                  .                   .                    .
```

Error Messages

Sometimes, Math cannot fit its answers into the format you have given. For instance, the answer may have six digits, but format only allows four. In these cases, Math writes out the answer in standard format, with a percent sign, %, in front.

This is the only form of error message that Math writes into the table. All other error messages appear at the bottom of the screen.

Math examines a table line by line, looking for errors. When it finds one, it stops, moves the text of the table up, and writes the error message on the bottom eight lines of the screen. The message is a brief description of the nature of the error. An error display looks like this:

```
ERROR -- (message)

(line in which error was found)
                                    ^
To continue: Press RETURN and correct the error.
```

The caret, ^, marks the place where Math found the error. It appears whenever the error is in a control block.

Suppose you enter <:D1> + <:D2> <:D3> = <:D4>. You will get this error message:

```
ERROR -- Formula in error at this position.

FORMULA <:D1> + <:D2> &
                      ^
To continue: Press RETURN and correct the error.
```

Here, the equation contains &, which is not an allowed character. Math does not display the equation beyond the position of the error.

Most errors, like the preceding sample, prevent Math from understanding what your commands mean or what numbers you wish to use. These are equivalent to errors of spelling or grammar in ordinary language.

NOTE: *The error is not always at the point where Math puts the caret. Sometimes, you can correct the error by changing something earlier in the line, or by changing something in a previous line.*

For example, look at this equation, which is missing a closing parenthesis, **)**:

```
(<:D1> + <:D2> * <:D3> = <:D4>
```

It will produce the following error message:

```
ERROR -- Formula in error at this position.


FORMULA   (<:D1> + <:D2> * <:D3> =
                                 ^


To continue: Press RETURN and correct the error.
```

You do not have to change the = sign to correct the equation. Instead, you need to put in a parenthesis, **)**, probably after **<:D2>**.

There are twenty-one possible error messages. Here is a list of them, with explanations and suggestions for making corrections:

```
Formula contains too many operations
```

The equation contains too many + and – signs. Take out as many as you can, or do your calculation in several short equations instead of one long one. For example, use: **<:A>** + **<:B>** = **<:C>** and **<:C>** + **<:D>** = **<:E>** instead of: **<:A>** + **<:B>** + **<D>** = **<:E>**.

```
Too many field names specified
```

An equation can only contain about 30 field names. Do your calculation in several short equations instead of one long one.

```
No room left to store this field name
```

The total length of all the field names, added together, is too great. Rewrite the equations with shorter names.

```
Control block equations contain too many terms
```

A control block can only contain about 40 field names. Split up your control block and do some of the calculations in another block, with another table.

```
Too many formulas specified
```

The total length of all your equations, added together, is too great. Split up your control block and do some of the calculations in another block, with another table.

```
Line has exceeded 200 characters
```

No command may have more than 200 characters. Use several short commands instead of one long one.

```
No room for this result: #########
```

######### stands for a number Math tried to write out. There is no room in the table for this number. Tab positions, margins, or text are in the way. Rearrange the table to provide more room.

```
No room for these results: #########
                     and: #########
```

Each ######### stands for a number Math tried to write out. One or the other number fits in the table, but not both. Tab positions, margins, or text are in the way. Rearrange the table to provide more room.

```
Undefined command at this position
```

The word following WPSMATH was something besides BEGIN, END, SET, FORMULA, TOTAL, TRUNCATE, or ROUND. These seven are the only commands allowed by Editor Math. They must be spelled correctly.

```
Second BEGIN command in Math area
```

A second BEGIN command appears in the Math area, after the first one. There should be only one BEGIN command in the area, and it must be the first command. Delete the extra BEGIN command.

```
BEGIN command does not follow WPSMATH
```

The BEGIN command must come directly after the word WPSMATH. It must not share the line with any other text. Edit the control block to put the BEGIN command in the proper place.

```
END command does not follow WPSMATH
```

The END command must come directly after the word WPSMATH. It must not share the line with any other text. Edit the control block to put the END command in the proper place.

```
Text in control block after END command
```

The END command must always be the last command in the control block. Edit the Math area to put the END command in the proper place, or delete the text that follows it.

```
Unmatched angle brackets
```

Every field name must have exactly one < at the beginning and exactly one > at the end. Brackets must appear nowhere else. Edit the line shown and remove the extra brackets or put in the missing ones.

```
Field name too long
```

A field name can have no more than 30 characters. Shorten the overlong name.

```
Divided by zero
```

Math divided by a field, which takes the value zero at some point. If the field value comes from a record, change or remove that record. If the field value comes from previous calculations, change the equations so that the field value never becomes zero.

```
Formula in error at this position
```

There are several possible errors. The error is in an equation. Make sure that:

- The characters + – * / and = are always followed by a field name, a constant, or text in parentheses, **( )**.
- All field names, constants, and pieces of text in parentheses, **( )**, are separated from their neighbors by + – * / or =.
- There is exactly one **)** for every **(**.

- There is a return at the end of every equation, or, if there is a format word, at the end of the format word.
- There is exactly one = sign in every FORMULA and SET command, with exactly one field name after it.
- The commands WPSMATH, TRUNCATE, and ROUND each have a line to themselves.
- Every field name starts with <: and ends with >.
- The only characters in constants are: **0 1 2 3 4 5 6 7 8 9 , + – .** and **$**.
- The characters **+ – . $** appear no more than once in any constant, in the correct place.
- Commas, **,**, do not appear at the beginnings of numbers.
- No constant has more than thirteen digits.

```
Formula in error at this position (unexpected return)
```

The command ends too soon. Make sure there is an = sign, a constant or complete field name to the left of the = sign, and a complete field name to the right of the = sign. Make sure the field name does not contain a hard return.

```
Format in error at this position
```

There are several possible errors. The error is in a format word. Make sure that:

- The only characters in format words are **– ( ) $ * , . 9.**
- The characters **– ( ) $ * .** appear no more than once.
- The **–** sign is either first or last, if it appears.
- The characters **( )** and the **–** sign do not appear in the same word.
- The characters **( )** come first and last, if they appear
- There is exactly one **)** for every **(** .
- The **$** sign comes before everything except **–** or **(** .
- The ***** comes before **9 , .)** or trailing **–** sign.

- There are no commas after the decimal point.
- There are no more than thirteen **9**s in the format word and no more than six **9**s to the right of the decimal point.
- Format words containing the characters * – . always contain **9**s.

`Number in error at this position`

The error is in a constant in a control block. Make sure that:

- The only characters in the constants are **0 1 2 3 4 5 6 7 8 9 $ * , . + – ( )**.
- There are no more than thirteen digits in the constant.

`Number in line is in error`

There are several possible errors. The error is in a number you entered in the table. Make sure that:

- There are no empty pairs of parentheses, **( )**.
- The only characters in the field value are **0 1 2 3 4 5 6 7 8 9 $ * , . + – ( )**.
- The characters **$. + – ( )** appear no more than once.
- The characters **$ * + (** only appear at the beginning.
- The character **)** only appears at the end.
- The **+** sign comes first, if it appears.
- The **–** sign is either first or last, if it appears.
- The **$** sign comes before * and after the **+** or **–** signs.
- There is exactly one **)** for every **(**.
- The characters **( )** come first and last, if they appear.
- The characters **( )** and the **–** sign do not appear in the same field value.

- There are no more than thirteen digits in the field value.
- The only characters directly under the decimal tab marker in the ruler are) . or one of the following invisible characters: tab, hard return, word-wrap return, paragraph marker, centering mark, end of document. You can check for the invisible characters by pressing Gold VIEW.

Summary of Editor Math

Editor Math is used to create numerical tables in documents. You supply part of the information on a line of the table, then Math calculates the rest of the information and completes the line for you.

In addition to doing calculations on separate lines, Math can total up columns, do subtotals, and calculate column averages.

By giving Math information for some calculations and not others, you can make it perform different calculations on different lines.

Math Areas and Math Control Blocks. DECmate turns Math on whenever you move forward into a Math area. It turns Math off when you move out of the Math area, use go-to-page, press Gold TOP DOCMT, or press Gold BOT DOCMT. Math will turn off if you back up into, out of, or within a Math control block.

Math areas begin and end with Math control blocks. There may be more control blocks inside the Math area. To create a Math control block, press Gold CMND and type WPSMATH. Then put in your Math commands. When you are done, press Gold CMND again.

Interactive Mode. When Editor Math is calculating, the *interactive mode* is on. To turn off the interactive mode and stop Math from calculating, go to the Editor Menu and type MA NO. Math will stay off until you turn it on again, or until you create a new document or select an old one for editing. To turn the interactive mode back on, go to the Editor Menu and type MA YES. The interactive mode will stay on until you turn it off.

Math Commands. There are seven possible Math commands: BEGIN, END, TOTAL, ROUND, TRUNCATE, SET, and FORMULA. Each command must have a separate line. To enter a command into a control block, type the command word as the first word of the line. BEGIN, END, TOTAL, ROUND, and TRUNCATE must not have any other text on the line with them. FORMULA and SET must have an equation on the line with them.

The first command in the first control block of a Math area must be BEGIN. It comes immediately after the word WPSMATH. The only command in the last control block of a Math area must be END. BEGIN, ROUND, SET, TOTAL, FORMULA, and TRUNCATE commands may share the same control block.

Tables. Editor Math is designed for creating numerical tables. These tables are organized by decimal-aligned tab positions. Do not use these tabs to position text; Math would label the text as an invalid number. Make sure there is enough room for the numbers you and Math will be writing in the table. If Math cannot fit its answers into the table, it will stop and give you an error message.

Tab Positions and Field Names. When Math calculates, it looks for information at decimal-aligned tab positions. It also writes the answers at decimal-aligned tab positions. These tab positions have field names, <:D1>, <:D2>, <:D3>, and so forth. The letter D must be capitalized. The field values for these names are the numbers written at the tab positions. The number in the field name is the number of the decimal-aligned tab position, going from left to right. Use these names in the equations of SET and FORMULA commands.

Totaling Tab Positions and Accumulators. Besides tab fields (named <:Dn> where n is the number of the column), there are two other kinds of fields you can use in equations. One is the totaling tab, named <:Tn>, where n is the number of the column you want the total written in. The letter T must be capitalized. The field values of <:Tn> fields are only written out on the line following a control block containing a TOTAL command.

The other kind of field is an accumulator. Its name is written <:nnn>, where nnn is any text, shorter than 30 characters, containing no return characters. Accumulator fields are never written out. They contain numbers you need in your calculations but do not want to see.

Allowed Characters. Aside from names in field names, the only characters allowed in an equation are **0, 1, 2, 3, 4, 5, 6, 7, 8, 9, +, –, *, /, $, (,), =,** comma, decimal point, and space. * stands for multiplication. / stands for division. Calculations inside **(** and **)** are performed first. Math ignores commas, spaces, and **$** signs; they serve only to make numbers more readable to you.

Writing Equations. In general, you should enter the equation just as if you were writing it down to work out on paper. There must be exactly one equation in every SET or FORMULA command. There must be exactly one = sign in every equation. The only thing in the equation that may follow the = sign is a field name. This is the field name of the tab or accumulator where you want Math to write the answer.

Math's Use of Equations. Math tries to use every FORMULA equation in a control block on every line of the table following the control block. If it does not have enough information, it writes no answer. If it comes to a new control block, it stops using the old set of equations. It will save the number in a total or accumulator field until a new number comes in. Always enter the equations in the order you would use them yourself, because *Math always performs the equations in the order you enter them.*

Totals. To calculate column totals, use a FORMULA command, which looks like this:

```
FORMULA <:Tn> + <:Dn> = <:Tn>
```

The n is the number of the column you wish to total up. At the bottom of the table, write a TOTAL command immediately above the line you want the total to appear on.

To total up the number of lines in your table, use a FORMULA command, which looks like this:

```
FORMULA <:nnn> + 1 = <:nnn>
```

<:nnn> is the name of the accumulator you are recording the total in. You may want to use a totalling tab (<:Tn>) instead of an accumulator, if you want the number written out.

SET Commands. Math uses a SET equation only once, at the time you enter it or scroll through it. Use SET commands for calculations you want done only once.

Averages. To calculate an average, use a SET or FORMULA command to divide the total by the number of entries, before you write it out.

Formats. Math always writes out its answers in standard format, unless a format word follows the equation in which the answer is calculated. In standard format, Math writes numbers without commas, asterisks, or dollar signs, indicates negative numbers by a leading minus sign, and writes out fractions to no more than six decimal places.

The format word tells Math how to vary the way the number is written. It begins and ends with quotation marks, " ". If the format word contains a dollar sign, **$**, Math writes the answer with a dollar sign. If the format word contains commas, Math writes the answer with commas every three places to the left of the decimal point. If you put a number of **9**s after the decimal point in the format word, Math will write out that many decimal places on fractions (up to six). If you put an asterisk, *****, in the format word, Math fills out the empty places in the answer with asterisks.

If you put parentheses, **( )**, or a trailing minus sign in the format word, Math uses them to indicate negative answers.

Truncation and Rounding Off. A TRUNCATE command makes Math drop the extra decimal places in a fractional answer, without rounding off the written part of the answer. There should be only one TRUNCATE command in a control block. It must have a line to itself.

A ROUND command cancels any previous TRUNCATE command. There should be only one ROUND command in a control block. It must have a line to itself. Math goes back to rounding automatically at the beginning of a new Math area.

4

List Processing Math

To use List Processing Math, you need some knowledge of Editor Math and List Processing. A great deal of List Processing Math is the same as Editor Math. Each uses control blocks and Math areas. FORMULA, SET, ROUND, and TRUNCATE commands are written in the same way. Equations and format words are the same.

When List Processing Math and Editor Math are not exactly the same, they are often very similar. Just as Editor Math uses its equations on every line of the table, List Processing Math uses its equations on every record of the list.

Editor Math writes its answers into the line it got its numbers from. List Processing Math puts its answers in the output document, along with the information from the list records.

The following chart summarizes the differences between Editor Math and List Processing Math.

Editor Math	**List Processing Math**
Uses BEGIN, END, and TOTAL commands.	*Does not* use BEGIN, END, or TOTAL commands.
Requires rulers with decimal tab positions.	*Does not* require special rulers.
Gets information from decimal-aligned tab positions.	Gets information from list records.
Uses all equations on every line of the table.	Uses all equations on every record of the list.
Control blocks usually appear at the top of the table.	Control blocks usually appear at the beginning of the list document.
Answers appear in the table.	Answers appear in the output document.
Reworks calculations whenever you scroll through the table.	Reworks calculations whenever you run List Processing with its list document.

List Processing Math works as part of the List Processing feature. All the usual rules of List Processing still apply. You still need a selection specification, a form document, and a list document.

Math Commands in the List Document

To use Math while list processing, you must put the Math commands in the list document. Put the commands in control blocks, like the ones used in Editor Math. The control blocks should be placed at the top of the list document, or between records.

Math performs the calculations in control blocks, using the data in the list records. The Math commands in a control block apply to all the records following them, until Math encounters another control block or until the document ends. Math commands can come before or after comments at the beginning of the list document.

Three Math commands are never used in List Processing: TOTAL, BEGIN, and END. If you use them, you will get an error message.

When you use Math in editing, the TOTAL command adds up all the numbers at a given tab. Since the information in List Processing is not organized by tab, you cannot use TOTAL commands. Other ways to calculate totals are described below, in "Totals and Averages."

Starting and Stopping Math. When you use Math in editing, BEGIN and END mark those parts of the document where you want to use Math. In list processing, Math starts working as soon as it comes to a Math control block in the list document; there is no need for a special BEGIN command.

Math stops working when it comes to the end of the list document, or when it comes to a control block with no FORMULA commands in it. Such a control block is called a *null control block.* A typical null control block looks like this:

```
----------------------- START CONTROL -----------------------
WPSMATH
SET    0 = <:field name 1>
SET    0 = <:field name 2>
        .
        .
        .
------------------------ END CONTROL ------------------------
```

To create a null control block, press Gold CMND and type WPSMATH. Use SET commands to put zero in every Math field in the form document. Then press Gold CMND again.

After the null control block, List Processing will write zeros in the form document, instead of numbers calculated by Math. If you do not set the form document's Math fields to zero, List Processing repeats the field values from the last calculated record, instead of writing out zeros.

Math will start calculating again when it comes to a control block with FORMULA commands in it.

Figure 4 shows the organization of a list document using Math control blocks.

COMMENTS (optional)
MATH CONTROL BLOCK
COMMENTS (optional)
RECORD
RECORD (The first control block uses these records.)
RECORD
MATH CONTROL BLOCK
RECORD
RECORD (The second control block uses these records.)
RECORD
NULL CONTROL BLOCK
RECORD
RECORD (Math does not use these records.)
RECORD
MATH CONTROL BLOCK
RECORD
RECORD (The third control block uses these records.)
RECORD
End of File

Figure 4 Locations of Math Commands in a List Document

Math Formulas in List Processing

Just as Editor Math looks for information at tabs, List Processing Math looks for information in the fields of records in the list document. These records use field names similar to the usual List Processing field names.

As a result, Math equations use these field names instead of decimal tab field names. For example:

```
FORMULA   <:stock> + <:new> = <:total>
FORMULA   <:cost each> * <:items> = <:cost all> "$999.99"
FORMULA   <:price> * 0.1 = <:tax> "$99.99"
FORMULA   <:price> + (<:price> * 0.1) = <:total> "$9999.99"
```

The names on the left side of the = signs are the names of fields in the records. The names on the right of the = signs (<:total>, <:cost all> and <:tax>) do not have to appear in the list records. But, if you want to know their values, those field names should appear in the form document. When the list document is merged with the form document, Math inserts its answers in place of the field names.

Notice that all of the names begin with a colon, : . The colon identifies the field as one that will be used in a Math calculation. All the field names used in Math must start a right angle bracket and colon, <:. They must not have a return in them. In all other ways, they are like ordinary field names.

You do not have to put a colon, : , in a field name if it is not going to be used in a Math calculation. If you put a colon in a field name that is *not* going to be used by Math, it does not affect either Math or List Processing.

Exercise 1: Making a Trial Run.

- Create the following list and form documents.
- Add your own entries after the third record of the list document.
- Use *process record* for the selection specification.
- Run List Processing.
- Send the output to a document; do not have it printed.

The form document:

```
Sales Representative Report

<!S><name>, working in <area>, made <:sales>. Expenses were
<:expenses> and the commission was <:commission>, leaving a
profit of <:profit>.

<!E>
```

Notice the field names in the form document. When List Processing produces the output document, it will get <name>, <area>, and <:sales> from the list records. But <:commission> and <:profit> are not in the records; List Processing gets those fields from Math.

The list document:

```
This is a list of all our sales representatives, their
sales for the month, their commissions, and their expenses.

----------------------- START CONTROL -----------------------
WPSMATH
FORMULA   <:sales> * 0.1 = <:commission> "$99,999.99"
FORMULA   <:sales> - <:commission> - <:expenses> = <:profit>
"$99,999.99"
------------------------ END CONTROL ------------------------


This list should be updated on the first Monday of
every month and the report sent to Hill and Armstrong.

<name>Willie Logan
<area>New York
<:sales>$8809.00
<:expenses>$550.78
<>

<name>Yvonne Leddy
<area>London
<:sales>$10,085.00
<:expenses>$803.45
<>

<name>Phil R. Bruschmann
<area>Amsterdam
<:sales>$5040.00
<:expenses>$478.09
<>
```

(Add your own entries.)

Notice that comments can come both before and after the Math control block. Notice also that <name> and <area>, which are not used in calculations, have no colon, : .

- When you finish, look at the output document. It should look like this:

The output document:

```
Sales Representative Report

Willie Logan, working in New York, made $8809.00.
Expenses were $550.78 and the commission was $880.90,
leaving a profit of $7,377.32.

Yvonne Leddy, working in London, made $10,085.00.
Expenses were $803.45 and the commission was $1,008.50,
leaving a profit of $8,273.05.

Phil R. Bruschmann, working in Amsterdam, made $5040.00.
Expenses were $478.09 and the commission was $504.00,
leaving a profit of $4,057.91.
```

(Your own entries follow.)

Exercise 2: More than One Control Block. The second exercise alters the Math commands part way through the list document. Suppose Mr. Logan and Ms. Leddy have their commission increased to twelve percent, while Mr. Bruschmann and the others remain at ten percent.

- Edit your list document to look like the one below. Use the CUT and PASTE keys to duplicate the Math control block, then change the value of <:cut> to 10.
- Use *process record* for the selection specification.
- Run List Processing.
- Add the output to the end of your first output document.

The list document:

```
This is a list of all our sales representatives, their
sales for the month, their commissions, and their
expenses.

----------------------- START CONTROL -----------------------
WPSMATH
SET 12 = <:cut>
FORMULA <:sales> * <:cut>/100 = <:commission>
"$99,999.99"
FORMULA <:sales> - <:commission> - <:expenses> = <:profit>
"$99,999.99"
------------------------ END CONTROL ------------------------

This list should be updated on the first Monday of
every month and the report sent to Hill and Armstrong.

<name>Willie Logan
<area>New York
<:sales>$8809.00
<:expenses>$550.78
<>

<name>Yvonne Leddy
<area>London
<:sales>$10,085.00
<:expenses>$803.45
<>

----------------------- START CONTROL -----------------------
WPSMATH
SET 10 = <:cut>
FORMULA <:sales> * <:cut>/100 = <:commission>
"$99,999.99"
FORMULA <:sales> - <:commission> - <:expenses> =
<:profit> "$99,999.99"
------------------------ END CONTROL ------------------------

<name>Phil R. Bruschmann
<area>Amsterdam
<:sales>$5040.00
<:expenses>$478.09
<>
```

(Your own entries follow.)

- When you finish, look at the output document. The new entries should look like this:

The output document:

```
Sales Representative Report

Willie Logan, working in New York, made $8809.00.
Expenses were $550.78 and the commission was $1,057.08,
leaving a profit of $7,201.14.

Yvonne Leddy, working in London, made $10,085.00.
Expenses were $803.45 and the commission was $1,210.20,
leaving a profit of $8,071.35.

Phil R. Bruschmann, working in Amsterdam, made $5040.00.
Expenses were $478.09 and the commission was $504.00,
leaving a profit of $4,057.91.
```

(Your own entries follow.)

Notice that the figures for commission and profit have changed in Logan and Leddy's records, but not in the rest.

Totals and Averages

You cannot use TOTAL commands in List Processing Math. Instead, you use FORMULA commands with accumulator fields. Like the accumulator fields in Editor Math, these may have any field name, so long as it begins with a colon, : .

Suppose that you want to keep track of how many sales representatives you have. You can do so by using this FORMULA command:

```
FORMULA   <:total sales reps> + 1 = <:total sales reps>
```

Every time Math processes a record, it adds one to <:total sales reps>, which is the accumulator field. Just as in Editor Math, the field value always starts at zero. You can use a similar command to total up the profit:

```
FORMULA   <:total profit> + <:profit> = <:total profit>
```

Then, if you wish to find the *average* profit from a sales rep, you can divide the second total by the first:

```
FORMULA   <:total profit> / <:total sales reps> = <:avg
profit> "$99,999.99"
```

To see the values in <:total profit>, <:total sales reps>, or <:avg profit>, you need only make a place for them in the trailer of the form document. However, even if you do not put them in the form document, Math will calculate their values.

Exercise: Totals and Averages in the Trailer.

- Edit your list and form documents to look like the ones below.
- Use *process record* for the selection specification.
- Run List Processing.
- Send the output to a document; do not print it out.

List document:

```
This is a list of all our sales representatives, their sales
for the month, their commissions, and their expenses.

----------------------- START CONTROL -----------------------
WPSMATH
SET   12 = <:cut>
FORMULA   <:sales> * <:cut>/100 = <:commission>
"$99,999.99"
FORMULA   <:sales> - <:commission> - <:expenses> = <:profit>
"$99,999.99"
FORMULA   <:total sales reps> + 1 = <:total sales reps>
FORMULA   <:total profit> + <:profit> = <:total profit>
"$99,999.99"
FORMULA   <:total profit> / <:total sales reps> = <:avg
profit>   "$99,999.99"
------------------------ END CONTROL ------------------------

This list should be updated on the first Monday of
every month and the report sent to Hill and Armstrong.
```

```
<name>Willie Logan
<area>New York
<:sales>$8809.00
<:expenses>$550.78
<>

<name>Yvonne Leddy
<area>London
<:sales>$10,085.00
<:expenses>$803.45
<>

<name>Phil R. Bruschmann
<area>Amsterdam
<:sales>$5040.00
<:expenses>$478.09
<>
```

(Add your own entries.)

Form document:

```
Sales Representative Report

<!S><name>, working in <area>, made <:sales>. Expenses were
<:expenses> and the commission was <:commission>, leaving a
profit of <:profit>.

<!E>Summary:
Our sales representatives produced a total profit of
<:total profit>. A sales representative's average profit was
<:avg profit>.
```

- Look at the output.

The field <:total sales reps> does not appear in the form document. Math calculates its value, then uses that value to calculate <:avg profit>.

Notice the order in which the FORMULA commands appear in the list document.

1. The first one calculates <:commission>.
2. The second uses the value of <:commission> to calculate <:profit>.

3 The third FORMULA command keeps count of the number of sales representatives.

4 The fourth one uses <:profit> to keep a running total of <:total profit>.

5 Finally, the last FORMULA command uses the two totals to calculate the average profit per sales representative.

Each equation builds on the results of the preceding ones. In general, you should enter equations in the order you would use them yourself if you were doing the calculation.

Notice the field names in the trailer of the form document. In ordinary List Processing, you cannot have fields in the trailer. With Math, you can put fields in, as long as they are accumulator fields. Fields from the list records cannot appear in the trailer.

Changing the Selection Specification

So far, all the exercises have used *process record* as the selection specification. All the records in the list are used.

In list processing, Math works only on the selected records. If a record is not selected, Math does not calculate with it, or reckon it into any totals or averages, or use it in any way.

As a result, you can use the same Math commands and the same list and form documents, but get different output documents by changing the selection specification.

Exercise: Two Reports from One List. In this exercise, you will use the following list and form documents to generate two reports, one on sales representatives with commissions of twelve percent, and one on those with commissions of ten percent.

- Create a new form document or edit your old one to look like the one below.

- Edit your list document to look like the one below. Since List Processing can only select on the basis of fields in the records, you must edit the records to include a field for the representative's commission. Name this field <:cut>.
- Create two new selection specifications, as shown below.
- Run List Processing twice, once with each of the two new selection specifications.
- Send the output to a document; do not have it printed.

Form document:

```
Sales Representative Report
<:cut> Percent Commission

<!S><name>, working in <area>, made <:sales>. Expenses were
<:expenses> and the commission was <:commission>, leaving a
profit of <:profit>.

<!E>Summary:
A sales representative's average profit was <:avg profit>.
```

List document:

```
This is a list of all our sales representatives, their sales
for the month, their commissions, and their
expenses.
---------------------- START CONTROL ----------------------
WPSMATH
FORMULA   <:sales> * <:cut> / 100 = <:commission>
"$99,999.99"
FORMULA   <:sales> - <:commission> - <:expenses> = <:profit>
"$99,999.99"
FORMULA   <:total sales reps> + 1 = <:total sales reps>
FORMULA   <:total profit> + <:profit> = <:total profit>
"$99,999.99"
FORMULA   <:total profit> / <:total sales reps> = <:avg
profit>   "$99,999.99"
----------------------- END CONTROL -----------------------

This list should be updated on the first Monday of
every month and the report sent to Hill and Armstrong.
```

```
<name>Willie Logan
<area>New York
<:sales>$8809.00
<:expenses>$550.78
<:cut>12
<>

<name>Yvonne Leddy
<area>London
<:sales>$10,085.00
<:expenses>$803.45
<:cut>12
<>

<name>Phil R. Bruschmann
<area>Amsterdam
<:sales>$5040.00
<:expenses>$478.09
<:cut>10
<>
```

(Your own entries follow.)

First selection specification:

```
if <:cut>=<12>
then process record
```

Second selection specification:

```
if <:cut>=<10>
then process record
```

Error Messages

Sometimes, Math cannot complete the calculation you have given it. For instance, the answer might be larger than 9,999,999,999,999. Math cannot write out such a large number. Instead, it writes out ***Ovr.flo***, meaning "overflow."

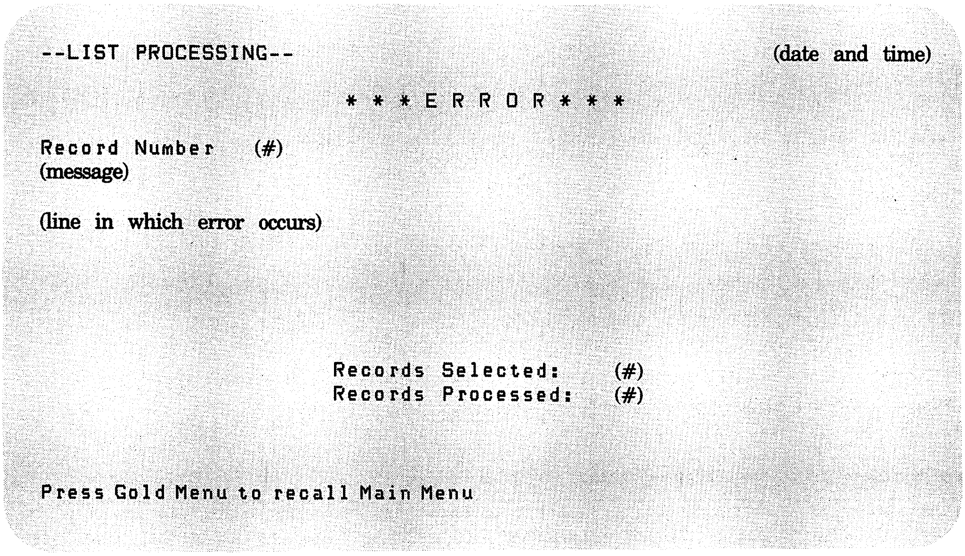
To get rid of this error, you must change the numbers you use or the SET FORMULA or commands, so that over-sized numbers do not appear.

Sometimes, Math does not have enough information in a record to complete a calculation. For instance, the calculation might call for <:profit> – <:tax>, but the tax field is empty. In that case, Math writes out ***Un.def***, meaning "undefined."

To get rid of this error, go back to the records and supply the missing information.

Most errors prevent Math from understanding what your commands mean or what numbers you wish to use. These are equivalent to errors of spelling or grammar in ordinary language.

When List Processing Math runs into an error, it stops processing and displays this message:



```
--LIST PROCESSING--                                      (date and time)

                          * * * E R R O R * * *

Record Number    (#)
(message)

(line in which error occurs)




                         Records Selected:    (#)
                         Records Processed:   (#)


Press Gold Menu to recall Main Menu
```

Figure 5 Error Message Screen

Record Number tells you which record the error is in. If the error is in a control block, the record number will be the number of the first record before the control block. If there are no records before the control block, the record number is 0.

The message is a brief description of the nature of the error. In some cases, Math sometimes displays the line containing the error below the message. The line ends at the place where Math could no longer make sense of the command or the number. The end of the line is marked with a caret, ^.

NOTE: The error is not always at the point where Math ends the line. Sometimes, you can correct the error by changing something earlier in the line, or by changing something in a previous line.

There are twenty-three possible error messages. Here is a list of them, with explanations and suggestions for making corrections:

```
Formula contains too many operations
```

The equation contains too many + and – signs. Take out as many as you can, or do your calculation in several short equations instead of one long one.

For example, use:

<:A> + <:B> = <:C> and **<:C> + <:D> = <:E>**

instead of:

<:A> + <:B> + <:D> = <:E>.

```
Too many field names specified
```

An equation can only contain about 30 field names. Do your calculation in several short equations instead of one long one.

```
No room left to store this field name
```

The total length of all the field names, added together, is too great. Rewrite the equations with shorter names.

```
Unmatched angle brackets
```

Every field name must have exactly one < at the beginning and exactly one > at the end. Brackets must appear nowhere else. Edit the line shown and remove the extra brackets or put in the missing ones.

```
Field name too long
```

The error is in a control block. A field name can have no more than 30 characters. Shorten the overlong name.

```
Field name exceeds 30 characters
```

The error is in a list record. A field name can have no more than 30 characters. Shorten the overlong name.

```
Field value number exceeds 30 characters
```

The error is in a list record. The number in a field cannot have more than 30 characters. Take out any unnecessary ones, such as commas and asterisks.

```
"<" within a field name
```

The error is in a list record. An extra < appears in a field name, such as <<:**sum**>. Remove the extra <.

```
">" within a field value
```

The error is in a list record. An extra > appears after a field name, such as <:**sum**>>. Remove the extra >.

```
Record not terminated with "<>"
```

The error is in a list record. Every record must end with <>. Put in a <> to end the record.

```
Text between records
```

Only records and control blocks are allowed in a list document, except for comments at the beginning. Move or delete the extra text, or edit it to make it into records or control blocks.

```
Record exceeds 2500 characters
```

No record may have more than 2500 characters. Shorten the overlong record.

```
Result document diskette full
```

The diskette you have selected has no room for your output. Start List Processing over again, but this time arrange for the output to go to a diskette with more room. You may delete documents from the current diskette, put a new diskette in the drive, or send the output to a different drive.

```
Control block equations contain too many terms
```

A control block can contain only about 40 field names. Split up your control block and do some of the calculations in another block, with a duplicate set of records.

```
Too many formulas specified
```

The total length of all your equations, added together, is too great. Split up your control block and do some of the calculations in another block, with a duplicate set of records.

```
Line exceeds 200 characters
```

No command may have more than 200 characters. Use several short commands instead of one long one.

```
Undefined command at this position
```

The word following WPSMATH was something besides FORMULA, SET, TRUNCATE, or ROUND. These four are the only commands allowed by List Processing Math. They must be spelled correctly.

```
Divided by zero
```

Math divided by a field that takes the value zero at some point. If the field value comes from a record, change or remove that record. If the field value comes from previous calculations, change the equations so that the field value never becomes zero.

`Formula in error at this position`

There are several possible errors. The error is in an equation. Make sure that:

- The characters + – * / or = are always followed by a field name, a constant, or text in parentheses, **( )**.
- All field names, constants, and pieces of text in parentheses, **( )**, are separated from their neighbors by + – * / or =.
- There is exactly one **)** for every **(** .
- There is a return at the end of every equation, or, if there is a format word, at the end of the format word.
- There is exactly one = sign in every FORMULA and SET command, with exactly one field name after it.
- The commands WPSMATH, TRUNCATE, and ROUND each have a line to themselves.
- Every field name starts with <: and ends with >.
- The only characters in constants are: **0 1 2 3 4 5 6 7 8 9 , + – . $**.
- The characters **+ – . $** appear no more than once in any constant, in the correct place.
- Commas, **,** , do not appear at the beginnings of numbers.
- No constant has more than thirteen digits.

`Formula in error at this position (unexpected return)`

The command ends too soon. Make sure there is an = sign, a constant or complete field name to the left of the = sign, and a complete field name to the right of the = sign. Make sure the field name does not contain a hard return.

`Format in error in this position`

There are several possible errors. The error is in a format word. Make sure that:

- The only characters in format words are: **– ( ) $ * , . 9**.
- The characters **– ( ) $ * .** appear no more than once.

- The – sign is either first or last, if it appears.
- The characters **( )** and the – sign do not appear in the same word.
- The characters **( )** come first and last, if they appear.
- There is exactly one **)** for every **(**.
- The **$** sign comes before everything except – or **(** .
- The * comes before **9 , .)** or trailing – sign.
- There are no commas after the decimal point.
- There are no more than thirteen **9**s in the format word and no more than six **9**s to the right of the decimal point.
- Format words containing the characters * – **.** always contain **9**s.

`Number within record is in error`

There are several possible errors. The error is in a field value. Make sure that:

- There are no empty pairs of parentheses, **( )**.
- The only characters in the field value are: **0 1 2 3 4 5 6 7 8 9 $** * **, . + – ( )**.
- The characters **$. + – ( )** appear no more than once.
- The characters **$** * **+ (** appear only at the beginning.
- The character **)** appears only at the end.
- The **+** sign comes first, if it appears.
- The – sign is either first or last, if it appears.
- The **$** sign comes before * and after the **+** or – signs.
- There is exactly one **)** for every **(**.
- The characters **( )** come first and last, if they appear.
- The characters **( )** and the – sign do not appear in the same field value.
- There are no more than thirteen digits in the field value.

```
Number in error at this position
```

The error is in a constant in a control block. Make sure that:

- The only characters in the constants are: **0 1 2 3 4 5 6 7 8 9 $ * , . + – ( )**.
- There are no more than thirteen digits in the constant.

Summary of List Processing Math

Math Commands. The control blocks, equations, and format words of List Processing Math are nearly identical to those of Editor Math. However, List Processing Math does not use the BEGIN, END, or TOTAL commands. Instead, any Math control block turns Math on. A null control block turns it off.

Math and List Processing. List Processing Math is used to do calculations in documents being produced by the List Processing feature. All the usual rules of List Processing apply. The Math control blocks are in the list document.

Math looks for information in the records following the control block. It writes the answers in the output document.

Field Names. <: should be the first two characters in the name of any Math field. Math ignores fields with no colon in their names. Math field names should have no more than 30 characters and should not contain return characters.

Math's Use of Equations. Math tries to apply every FORMULA command in the control block to every record following the block, until it comes to another control block or to the end of the document. When it comes to a new control block, it stops using all the equations from the first control block. Math keeps the information in old accumulator fields until it comes to new information under the accumulator's field name.

Whenever you run List Processing, Math recalculates the answers. You may change the records in the list document and the answers in the next version of the output document will still be accurate.

Totals. Since there is no TOTAL command in List Processing Math, you must calculate totals with accumulator fields instead of totaling tabs. Use a SET and a FORMULA command:

```
SET    0 = <:nnn>
FORMULA    <:nnn> + <:mmm> = <:nnn>
```

<:nnn> is the name of the accumulator you store the total in, and <:mmm> is the name of the field value you are totaling up.

You may total up the number of entries and calculate averages in List Processing Math, just as you did in Editor Math. Math can write out averages and totals in the trailer of the output document, if you provide field names for them in the form document.

Math calculates only with the selected records. If you change your selection specification, Math uses different records and gives different answers, including totals and averages.

A

Symbol Table

<:D#>	Decimal tab field names, used in Editor Math, in the equations of FORMULA and SET commands. If the field name follows an = sign, and if there is sufficient data, the contents of the field is printed out on every line of a table. (See pages 13, 29, 34, 58, 64, 67 to 68, 92.)
<:T#>	Total field names, used in Editor Math, in the equations of FORMULA and SET commands. The contents of total fields are printed out on the line following a control block with a TOTAL command. (See pages 29 to 30, 65, 96.)
<:name>	Accumulator field names, used in Editor and List Processing Math. Editor Math cannot print out accumulator field names. List Processing Math uses nothing but accumulator field names. (See pages 35, 64, 75.)
0, 1, 2, 3, 4, 5, 6, 7, 8, 9	Numerals. They are the commonest characters in Math's input and output. They can also be used within a control block, as constants. (See page 14.)
9	Nines, used in format words to show the highest number allowed as the answer to a given equation. (See pages 44, 65.)

“ ”	Quotation marks, used to mark the beginnings and ends of format words. (See pages 44, 66.)
.	Decimal point, used to write numbers for input and output, and for constants in control blocks. (See pages 14, 65.)
$	A dollar sign, used to indicate money in constants, format words, input, or output. (See pages 14, 44 to 47, 65, 66.)
+	A plus sign, used in equations, constants, or input to indicate addition or positive numbers. (See pages 14, 15, 65.)
–	A minus sign, used in equations, constants, format words, input, or output to indicate subtraction or negative numbers. (See pages 14, 19, 23, 48, 65, 66.)
*	An asterisk, used in equations to indicate multiplication, and used as a space-filler in input, output, and format words. (See pages 14, 26, 47, 65, 66.)
/	A slash, used in equations to indicate division. (See pages 14, 27, 65.)
( )	Parentheses, used in equations to group operations together, and used in format words, input, and output to indicate negative numbers. (See pages 14, 40 to 42, 50, 65, 66.)
=	An equals sign, used in equations to indicate the end of the calculation. Every equation must have exactly one equals sign in it. Exactly one field name must follow the equals sign. (See pages 14, 65.)
%	A percent sign. Math writes a percent sign before numbers it cannot write out in the format indicated by the format word. (See pages 45, 57.)
	A space. You can put spaces into the FORMULA command wherever you please. Math ignores them, but they can make the equation easier for you to read. Do not put extra spaces in a field name, or Math will not recognize it. Do not use spacing in place of tab stops. (See pages 11, 14, 44, 65.)

Glossary

Italicized words also appear in the glossary.

accumulator

A small section of memory in DECmate. *Editor Math* uses accumulators to store numbers that you need to calculate but do not want written out as answers (for example, the number of lines in the table, used in calculating an average). Accumulators are referred to by their *field names.*

BEGIN command

The *Math command* which starts a *Math area* in *Editor Math.* The BEGIN command must be the first command in the first *control block* of the Math area. It may share the same control block with *SET, FORMULA, ROUND*, and *TRUNCATE commands.*

command

See *Math command.*

constant

A number that you write into a Math *equation*. Constants keep the value you set them at; they do not change value from one line or record to the next, as do *fields*. Math uses the same constant every time it performs the calculation in which the constant appears.

control block

See *Math control block*.

decimal fraction

A fractional number written out in decimal form. The number 2.5 is the number two-and-one-half, written as a decimal fraction. Math writes all fractional numbers as decimal fractions.

decimal-aligned tab position

The column where decimal points are aligned when text is entered. It is marked by a decimal point in the ruler. *Editor Math* takes its information from decimal-aligned tab positions and writes its answers at them. They are referred to by their *field names*.

Editor Math

The version of Math that you can use while creating or editing a document. Editor Math completes numerical tables on the basis of the information you enter in the tables. It can also be used for single-line calculations.

END command

The command that finishes a *Math area*. The END command must be the last command in the last *control block* of the Math area. It must have a control block to itself.

equation

The description of a calculation that follows a *FORMULA* or *SET command.* It is made up of numbers, arithmetic symbols, and field names.

field

A section of memory in DECmate. Math stores all its numbers in fields. Fields may get their numbers from tab positions, list records, or calculations by the Math feature.

field name

A label Math uses for its fields. All field names used by Math begin with <: and end with >. Fields in List Processing records, *decimal-aligned tab positions*, *totaling tab positions*, and *accumulators* are referred to by field name.

format word

A word following the *equation* in a *FORMULA* or *SET command*, causing Math to write its answers in a manner different from *standard format.* Format words must begin and end with quotation marks.

FORMULA command

A command telling Math to perform a given calculation on every line of a table, or every record of a list. The particular calculation is described in the *equation* following the word FORMULA.

interactive mode

A manner of operation for *Editor Math* in which Math performs the calculations for each line as soon as you press RETURN. When interactive mode is off, Editor Math does nothing. To turn the interactive mode off (or on), go to the Editor Menu and type MA NO (or YES).

leading minus sign

A – sign on the left side of a number, indicating that the number is *negative*. The number –24 has a leading minus sign.

leading zero

A zero written before the decimal point of *a decimal fraction* which is smaller than one. The number 0.25 has a leading zero; the number .25 does not.

List Processing Math

The version of Math you can use with the List Processing feature. List Processing Math does calculations on the basis of information in the records of a list document. It writes its answers in the output document.

Math area

A section of text containing information used by *Editor Math* to perform its calculations and in which Math writes its answers. This text is usually in the form of *control blocks* and numerical tables.

Math command

Text in a *Math control block*, between one return character and another. There are seven legal Math commands: BEGIN, END, TOTAL, SET, FORMULA, ROUND, and TRUNCATE. Only the last four, FORMULA, SET, ROUND, and TRUNCATE, are used in *List Processing Math.* The words FORMULA and SET should be followed by an equation. The words BEGIN, END, TOTAL, ROUND, and TRUNCATE should be followed by a return.

Math control block

The text between a START CONTROL line and an END CONTROL line, beginning with the word WPSMATH. This text turns the Math feature on and off, and specifies the calculations it performs.

negative number

A number lower than zero. For instance, a debt is a negative amount of credit.

null control block

A *control block* in *List Processing Math,* containing no *FORMULA commands.* A null control block turns off List Processing Math.

operation

The act of addition, subtraction, multiplication, or division, represented in an *equation* by the symbols +, –, *, or /.

ROUND command

Math always rounds off fractional answers that it cannot write out in full, until it comes to a *TRUNCATE command.* The ROUND command cancels any previous TRUNCATE command and makes Math go back to *rounding off.* All the answers from a given control block are either rounded off or truncated.

rounding off

Writing out a *decimal fraction* in shortened form, in such a way that it is as nearly equal to its longer form as possible. If you round off 3.012 to two decimal places, you get 3.01. If you round off 3.017 to two decimal places, you get 3.02. 3.05 rounds off to 3.1. See *ROUND command.*

SET command

A command telling Math to perform a given calculation once, before it starts working on the following table or list of records. The particular calculation is described in the *equation* following the word SET.

standard format

The way Math writes out its answers, unless otherwise instructed in a *format word*. In standard format, Math uses no commas, **$** signs, or asterisks. It indicates *negative numbers* by a *leading minus sign*, uses a *leading zero*, and gives *decimal fractions* to six decimal places, without *trailing zeros*.

tab field name

A *field name* used by *Editor Math* to label the field holding numbers found at a given *decimal-aligned tab position*. It is written <:Dn>, where n is the number of the column in the table. If the number is at a *totaling tab position*, the name is written <:Tn>. The letters D and T must be capitalized.

TOTAL command

A command telling *Editor Math* to write out any column totals it has stored in the fields of *totaling tab positions*.

total line

The line just after a TOTAL command, where *Editor Math* writes out column totals.

totaling tab position

A *decimal-aligned tab position* at which a column total is written. See *TOTAL command*.

trailing minus sign

A – sign on the right side of a number, indicating that the number is negative. The number 24– has a trailing minus sign.

trailing zeros

Zeros following the last non-zero digit of a decimal fraction. The number 2.400000 has trailing zeros. Math does not write trailing zeros unless the *format word* calls for them.

TRUNCATE command

A command telling Math to *truncate* decimal fractions it cannot write out in full. After a TRUNCATE command, Math truncates all the fractional answers it cannot write out in full, until it comes to a *ROUND command* or the end of the *Math area.* All the answers from a given control block are either truncated or rounded off.

truncating

Writing out a decimal fraction in shortened form, leaving off some of the last digits. The number 3.018 and the number 3.012, truncated to two decimal places, are both 3.01. See *TRUNCATE command.*

whole number

A number with no fractional part. The numbers 1, 0, 38, and –20 are all whole numbers. The numbers 0.25, –9.1, 807.21, and 0.003 are not whole numbers.

WPSMATH

The first word in any *Math control block.* It must have a line to itself. The Math feature will not turn on if this word does not appear in the control block.

Index

- ^, 57 to 58, 80
- “, 44, 66, 90
- $, 14, 44 to 47, 65, 66, 90
- %, 45, 57, 90
- ( ), 14, 40 to 42, 50, 65, 66, 90
- *, 14, 26, 47, 65, 66, 90
- +, 14, 15, 65, 90
- ,, 14, 46 to 47, 65, 66
- –, 14, 19, 23, 48, 65, 66, 90
- ., 14, 65, 90
- /, 14, 27, 65, 90
- 9, 44, 65, 89
- :, 13, 14, 34, 71
- =, 14, 65, 90

A

- Accumulators, 35 to 40, 64, 75, 88, 91
  - description of, 35, 64, 91
  - field names for, 35, 64, 75
- Activating math, 2, 3 to 7
- Addition
  - combined with subtraction, 20 to 22
  - simple, 15 to 17
- Allowed Characters, *see Characters, allowed in Math equations*
- Angle brackets, 13, 34, 60, 64, 65, 71, 83, 87
- Areas, *see Math areas*
- Asterisks, 47, 90
- Averages, 1–2
  - in Editor Math, 38 to 40, 66
  - in List Processing Math, 75 to 78, 88

B

- BEGIN command, 10, 64, 91, 94
  - in error messages, 59
  - not in List Processing Math, 68 to 69, 87

C

- Calculations for specific lines, *see Specific lines, calculations for*
- Caret, 57 to 58, 80
- Characters, allowed in Math equations, 14 to 15, 65
- Circumflex, *see Caret*
- Colon, 13, 14, 34, 71
- Columns, order of, 21 to 22

Commands, *see Math commands*
Commas, 14, 46 to 47, 65, 66
Constants, 14, 24 to 26, 60 to 61, 85
Control blocks
  creating, 9, 63
  description of, 9 to 12, 37, 63

D

Decimal-aligned tab position, 13 to 18, 29, 34, 58, 64, 67 to 68, 92
  field name for, 13, 29, 34, 58, 64, 67 to 68, 92
Decimal fractions, 29, 43 to 45, 47, 92
Decimal point, 14, 65, 90
Division, 28 to 29
Division by zero, 29, 60, 84
Division sign, 14, 27, 65, 90
Dollar sign, 14, 44 to 47, 65, 66, 90

E

Editor Math
  compared to List Processing Math, 67 to 68
  general description of, 1, 9, 92
  summary of, 63 to 66
END command, 10, 60, 64, 67 to 68, 92
Equals sign, 14, 65, 90
Equations, 12, 14 to 15, 58 to 60, 65, 87, 93
Error messages, 11, 14, 17, 29, 45
  for Editor Math, 57 to 63
  for List Processing Math, 80 to 87

F

Feature diskette, 7
Field, 55, 60, 64 to 65, 89, 93
  for accumulators, 34 to 35
  for decimal-aligned tab positions, 13
  in List Processing Math, 70 to 71, 75, 77, 81 to 84, 87, 88
  for totaling tab positions, 29 to 30
Field name, *see also Tab Field Name,* 29, 55, 70, 77, 93
  for accumulators, 34, 58, 60, 64 to 65, 75, 82 to 83, 85, 87, 89
  for decimal-aligned tab positions, 13, 64, 89, 92
  for totaling tab positions, 64 to 65, 89, 96
Format words, 44
Form document, 68, 71 to 72, 77 to 79
FORMULA command, 89, 93
  creating, 11 to 13
  description of, 11 to 13
  in Editor Math, 11 to 13, 58 to 61, 64 to 65
  in List Processing Math, 67, 69 to 71, 80, 84 to 85, 87 to 88

I

Interactive Mode, 11, 63, 93

L

Largest possible number, 43
Leading minus sign, *see also Trailing minus sign, Parentheses*, 23 to 24, 43, 48, 50, 66, 94
Leading zero, 47, 94
List document, 68 to 74, 76, 79, 83, 87, 94

List Processing Math
  compared to Editor Math, 67 to 68
  general description of, 1–2, 67 to 68, 94
  summary of, 87 to 88

M

Math areas, 52, 59, 66, 94
  creating, 9 to 10, 63
  description of, 9 to 10, 63, 94
Math commands, 9, 59, 64, 84, 87, 94
  in Editor Math, 9, 64
  in List Processing Math, 67 to 68, 87
Math control blocks, *see Control blocks*
Math, general description of, vi, 1–2
Minus sign, *see also Leading minus sign, Trailing minus sign,* 14, 19, 23, 48, 65, 66, 90
Money format, 44 to 45, 52, 66
Multiple total columns, 32 to 33
Multiplication, 25 to 27

N

Negative numbers, 23 to 24, 48 to 51, 66, 95
Nines, 44, 66, 89
Null control block, 69, 87, 95
Numbers, 23, 29, 43
  largest possible, 43
  smallest possible, 43

O

One-column totals, 30 to 31
Operation, 58, 82, 95
Order of columns, *see Columns, order of*
Output document, 67 to 68, 87 to 88
  in examples, 71 to 73, 75 to 76, 78
Overflow (*Ovr.flo*), 80

P

Parentheses, 14, 50
  in format words, 50, 60 to 62, 66, 90
  in FORMULA equations, 40 to 42, 60 to 62, 65, 90
  for negative numbers, 50, 66, 90
Percent sign, 45, 57, 90
Plus sign, 14, 15, 65, 90

Q

Quotation marks, 44, 60, 90

R

ROUND command, *see also TRUNCATE command,* 52 to 54, 66, 95
Ruler
  in Editor Math, 13, 15, 18, 32
  in List Processing Math, 68

S

Selection specification, 71, 73, 76, 78 to 79
SET command, 35 to 37, 66, 95
Slash, *see Division sign*
Smallest possible number, 43
Space in Math commands, 11, 14, 44, 65
Specific lines, calculations for, 54 to 56

Standard format, 43, 66, 93, 96
Subtotals, 34 to 37
Subtraction, 19 to 20
Summary
  of Editor Math, 63 to 66
  of List Processing Math, 87 to 88
Symbol table, 89 to 90

T
Tab field name, 34, 58, 89, 93, 96
  for decimal-aligned tab positions, 13, 64 to 65
  not in List Processing Math, 67 to 68
  for totaling tab positions, 29 to 30, 65
Tables, 9, 12, 15
  creating, 15 to 19
  use of rulers in, 13, 15, 17 to 19, 30 to 32
Tab position, *see Totaling tab position, Decimal-aligned tab position*
TOTAL command, 29, 68 to 69, 75, 89, 96
Totaling tab position, *see also Totals,* 29 to 30, 34, 38, 65, 89, 96
  field name for, 29 to 30, 65, 89, 96
Total line, 30 to 32
Totals, *see also Totaling tab positions, One-column totals, Multiple total columns, Subtotals,* 29 to 39, 65, 75 to 78, 88
  in Editor Math, 29 to 39, 65
  in List Processing Math, 75 to 78, 88
Trailer, totals and averages in, 76 to 78
Trailing minus sign, *see also Leading minus sign, Parentheses,* 49, 66, 96
Trailing zeros, 47, 97
TRUNCATE command, *see also ROUND command,* 52 to 54, 66, 97

U
Undefined (*Un.def*), 81

W
Whole number, 43, 97

Z
Zero, 29, 47, 60, 84, 94, 97
  division by, 29, 60, 84
  leading, 47, 94
  trailing, 47, 97

HOW TO ORDER ADDITIONAL DOCUMENTATION

DIRECT TELEPHONE ORDERS

In Continental USA and Puerto Rico call **800–258–1710**	In Canada call **800–267–6146**	In New Hampshire, Alaska or Hawaii call **603–884–6660**

DIRECT MAIL ORDERS (U.S. and Puerto Rico*)

DIGITAL EQUIPMENT CORPORATION
P.O. Box CS2008
Nashua, New Hampshire 03061

DIRECT MAIL ORDERS (Canada)

DIGITAL EQUIPMENT OF CANADA LTD.
940 Belfast Road
Ottawa, Ontario, Canada K1G 4C2
Attn: A&SG Business Manager

INTERNATIONAL

DIGITAL EQUIPMENT CORPORATION
A&SG Business Manager
c/o Digital's local subsidiary
or approved distributor

Internal orders should be placed through the Software Distribution Center (SDC), Digital Equipment Corporation, Northboro, Massachusetts 01532

*Any prepaid order from Puerto Rico must be placed
with the Local Digital Subsidiary:
809–754–7575

DECmate II
Using Math
AA–N186A–TA

Reader's Comments

Your comments and suggestions are welcome. They will help us in our continuous effort to improve the quality and usefulness of our publications.

1 How would you rate this manual for:

	good	fair	poor
completeness of information	______	______	______
accuracy of information	______	______	______
ease of use (clarity, organization)	______	______	______

2 Did you find errors in this manual? Please specify by page and paragraph, or mark corrections on pages and attach copies to this form.

	page	paragraph
a) Incorrect information:	______	______
b) Information left out:	______	______
c) Hard to understand:	______	______

3 What suggestions do you have for improving this manual?

____

____

4 What parts of the manual were especially good at helping you understand things?

____

____

5 Please check the boxes that apply to you.

☐ Technical
☐ Nontechnical
☐ Management
☐ Nonmanagement
☐ Other (explain)____________________

☐ Used other word processors
☐ Used computers
☐ Use the word processing unit yourself

Name____________________Title____________________

Company____

Street____________________City____________________

State__________Zip____________________Date____________________

Do Not Tear - Fold Here and Tape

digital

No Postage
Necessary
if Mailed in the
United States

BUSINESS REPLY MAIL
FIRST CLASS PERMIT NO.33 MAYNARD MASS.

POSTAGE WILL BE PAID BY ADDRESSEE

ATTN: Commercial Engineering Publications MK1-2/E06
Office Products Documentation Group
DIGITAL EQUIPMENT CORPORATION
CONTINENTAL BOULEVARD
MERRIMACK, N.H. 03054

Do Not Tear - Fold Here and Tape

Cut Along Dotted Line

Cut Along Dotted Line

Cut Along Dotted Line

Printed in U.S.A